

Mini-Project Report On

Advanced AI Voice Assistant Integrated with CHATGPT

*Submitted in partial fulfillment of the requirements for the
award of the degree of*

Bachelor of Technology

in

Computer Science & Engineering

By

Govind Kiran (U2003088)

Nafia Basheer (U2003139)

Mohammed Nihal (U2003136)

Misha Mathew (U2003135)

Under the guidance of

Ms. Jisha Mary Jose



**Department of Computer Science & Engineering
Rajagiri School of Engineering and Technology (Autonomous)
Rajagiri Valley, Kakkanad, Kochi, 682039**

July 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039



CERTIFICATE

*This is to certify that the mini-project report entitled "**Advanced AI Voice Assistant Integrated with CHAT GPT**" is a bonafide work done by **Mr. Govind Kiran (U2003088)**, **Ms. Nafia Basheer (U2003139)**, **Mr. Mohammed Nihal (U2003136)**, **Ms. Misha Mathew (U2003135)**, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2022-2023.*

Dr. Preetha K. G.
Head of Department
Dept. of CSE
RSET

Mr. Uday Babu P.
Mini-Project Coordinator
Asst. Professor
Dept. of CSE
RSET

Ms. Jisha Mary Jose
Mini-Project Guide
Asst. Professor
Dept. of CSE
RSET

ACKNOWLEDGEMENTS

We wish to express our sincere gratitude towards **Dr. P. S. Sreejith**, Principal of RSET, and **Dr. Preetha K. G.**, Head of Department of Computer Science and Engineering for providing us with the opportunity to undertake our mini-project, "Unify App".

We are highly indebted to our mini-project coordinators, **Mr. Uday Babu**, Assistant Professor, Department of Computer Science and Engineering, **Ms. Tripti C.**, Assistant Professor, Department of Computer Science and Engineering for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our mini-project guide **Ms. Jisha Mary Jose**, for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Govind Kiran

Nafia Basheer

Misha Mathew

Mohammed Nihal Sageer

ABSTRACT

Voice assistants have become an integral part of our daily lives, providing convenience and assistance in various tasks. In this paper, we introduce a voice assistant powered by the state-of-the-art language model ChatGPT. Our voice assistant aims to enhance user interactions by providing accurate and natural language responses generated by ChatGPT, leveraging its vast knowledge and conversational capabilities. Through advanced natural language processing techniques, our voice assistant seamlessly converts spoken queries into textual inputs for ChatGPT, enabling a smooth conversational experience. The underlying ChatGPT model, trained on diverse and extensive textual data, offers a broad understanding of human language and a wide range of knowledge spanning multiple domains. To ensure optimal performance, our voice assistant employs sophisticated speech recognition and voice synthesis technologies, enabling users to interact with it effortlessly and receive responses in a human-like voice. The assistant's intuitive user interface allows for seamless integration into various devices, empowering users to engage in natural and productive conversations. In conclusion, our voice assistant powered by ChatGPT opens up new possibilities for seamless and intelligent conversational experiences. With its ability to comprehend and generate natural language responses, the assistant brings forth a valuable tool for information retrieval, task automation, and personalized assistance, ultimately enhancing user productivity and convenience in various domains.

Contents

Acknowledgements	ii
Abstract	iii
List of Figures	vii
1 Introduction	1
1.1 Background	1
1.1.1 Evolution of AI Voice Assistants:	1
1.1.2 ChatGPT	2
1.1.3 Integration of AI Voice Assistants with ChatGPT:	2
1.2 Existing System	3
1.3 Problem Statement	4
1.4 Objectives	4
1.5 Scope	4
2 Literature Review	6
2.1 SARA: A Voice Assistant Using Python.	6
2.2 Voice assistant integrated with chat gpt	6
2.3 The voice enabled personal assistant for PC using python.	7
2.4 Comparison	8
3 System Analysis	10
3.1 Expected System Requirements	10
3.2 Feasibility Analysis	10
3.2.1 Technical Feasibility	10
3.2.2 Operational Feasibility	10
3.2.3 Economic Feasibility	11
3.3 Hardware Requirements	11

3.4	Software Requirements	11
3.4.1	Visual Studio Code (Python(ver.3.11))	11
4	Methodology	12
4.1	Explanation of each block	12
4.1.1	User's Voice Command	12
4.1.2	Speech to Text API	12
4.1.3	ChatGPT API:	12
4.1.4	Voice Assistant Output (Textual):	13
4.2	Methodology for Implementing the ChatGPT API Voice Assistant:	13
4.2.1	Model Selection:	13
4.2.2	API Integration:	13
4.2.3	Error Handling:	13
4.2.4	Personalization and Context Retention:	14
4.2.5	User Interface:	14
4.2.6	Testing and Optimization:	14
4.2.7	Security and Privacy:	14
5	System Design	16
5.1	Architecture Diagram	16
5.2	Sequence diagram	17
5.2.1	Login	17
5.2.2	Speech to Text	17
5.2.3	OpenAPI connection	17
5.2.4	Text to Voice	18
5.2.5	Remainder	18
6	System Implementation	19
6.1	Login	19
6.2	Speech to Text	19
6.3	OpenAPI connection	20
6.4	Text to voice	21
6.5	Remainder	22

7	Testing	24
7.1	Unit Testing:	24
7.2	Integration Testing:	24
7.3	Functional Testing:	24
7.4	User Acceptance Testing (UAT):	25
8	Results	26
9	Risks and Challenges	29
10	Conclusion	30
	References	31
	Appendix A: Base Paper	31
	Appendix B: Sample Code	51
	Appendix C: CO-PO and CO-PSO Mapping	67

List of Figures

4.1	Block Diagram	15
5.1	Architecture diagram	16
5.2	Login	17
5.3	Speech to Text	17
5.4	OpenAPI connection	18
5.5	Text to Voice	18
5.6	Task remainder	18
8.1	Login page	26
8.2	Main menu	26
8.3	Users Page	27
8.4	AI Assistant interface	27
8.5	First input: write a letter	27
8.6	Output: writing letter	27
8.7	Response saved as audio file	28
8.8	OCD assistance	28

Chapter 1

Introduction

1.1 Background

1.1.1 Evolution of AI Voice Assistants:

The inception of AI voice assistants can be traced back to the early 2000s when the first voice-activated systems emerged. These early iterations were largely rule-based, using specific keywords and patterns to trigger predefined responses. However, with advancements in machine learning and natural language processing, voice assistants underwent significant improvements. The introduction of statistical-based models, such as Hidden Markov Models (HMMs) and Gaussian Mixture Models (GMMs), enabled more accurate speech recognition and basic natural language understanding. This laid the foundation for popular voice assistants like Apple's Siri and Google Assistant, which rapidly gained prominence with the proliferation of smartphones and smart speakers.

Despite their widespread adoption, current voice assistants face several limitations. One of the main challenges lies in understanding complex and contextually rich queries. Traditional rule-based systems struggle with varied sentence structures and nuances in natural language, leading to inaccurate interpretations and limited scope in handling user inputs. Moreover, retaining context during multi-turn interactions remains a challenge, often resulting in repetitive responses and a lack of continuity in conversations. The limited integration of advanced language models hinders the generation of creative and dynamic outputs, preventing voice assistants from delivering truly engaging and human-like interactions. As a result, users may find their interactions with these assistants somewhat mechanical and unsatisfying.

1.1.2 ChatGPT

In recent years, transformer-based language models have sparked a revolution in natural language understanding and generation. ChatGPT, based on the GPT-3.5 architecture, is one such language model that has garnered widespread attention. Trained on a diverse and extensive dataset, ChatGPT exhibits an impressive ability to generate contextually relevant and coherent text, mimicking human-like responses. Its versatility has been demonstrated in various domains, from chatbots and customer support to creative writing and code completion. ChatGPT's capabilities open new possibilities for enhancing conversational experiences and addressing the limitations of existing voice assistants.

1.1.3 Integration of AI Voice Assistants with ChatGPT:

The concept of integrating AI voice assistants with ChatGPT has emerged as a promising approach to enhance the sophistication of virtual assistants. By combining the strengths of voice recognition and ChatGPT's language modeling, this integration aims to create more interactive and context-aware conversational experiences. The process involves converting user speech into text using speech recognition systems and then utilizing ChatGPT to generate text-based responses. This seamless integration seeks to bridge the gap between spoken language and written text, enabling more natural and dynamic interactions.

One of the key benefits of integrating ChatGPT with voice assistants is the potential for advanced natural language understanding. ChatGPT's ability to comprehend the context and nuances of user queries can significantly improve the accuracy of speech recognition and enhance the overall conversational experience. The virtual assistant's increased understanding of user inputs facilitates more accurate intent recognition and ensures that the responses are contextually relevant and coherent.

Another advantage of this integration lies in the retention of context during multi-turn interactions. ChatGPT's capacity to remember past queries and responses allows the virtual assistant to maintain continuity in conversations and provide more contextually appropriate answers. Additionally, by leveraging historical user interactions, the system can offer personalized responses tailored to individual preferences and requirements. This personalization enhances user satisfaction and fosters a sense of connection with the virtual assistant.

While integrating ChatGPT with voice assistants offers numerous benefits, it also raises ethical considerations. As ChatGPT can generate content, there is a need to ensure responsible use and mitigate potential risks, such as misinformation and bias. Safeguarding user privacy and data security is of utmost importance, given the involvement of voice data processing and storage. Efforts must be made to address these ethical concerns and adhere to best practices in AI development.

In conclusion, the integration of AI voice assistants with ChatGPT represents a significant advancement in the field of virtual assistants. By combining voice recognition with advanced language modeling, this integration seeks to create more interactive, personalized, and engaging conversational experiences. Addressing the limitations of current voice assistants and building upon the capabilities of ChatGPT, this project holds the promise of reshaping human-computer interactions and elevating the virtual assistant landscape to new heights.

1.2 Existing System

- Amazon Alexa: Developed by Amazon, It can perform a wide range of tasks, including setting reminders, playing music, answering questions, controlling smart home devices, and more.
- Google Assistant: Developed by Google,. It is integrated into various devices, and can provide personalized responses based on user interactions and preferences.
- Apple Siri: Siri is Apple's AI voice assistant.It can handle tasks such as sending messages, making phone calls, setting reminders, and providing information using natural language processing.
- Microsoft Cortana: Cortana is Microsoft's AI voice assistant. It can assist with tasks like setting reminders, scheduling appointments, and providing personalized recommendations.
- Samsung Bixby: Bixby is Samsung's voice assistant, integrated into Samsung smartphones and other devices. It offers voice control for various applications, device settings, and smart home devices.

1.3 Problem Statement

The challenge is to create a voice assistant that can answer almost every question and provides with features to save pdf and mp3 files, also with additional features like pronunciation, storytelling, rapping etc.

The proposed system aims to bridge the gap between spoken language and written text, allowing for more natural and interactive user interactions. The project will explore the creative capabilities of ChatGPT to generate dynamic and contextually relevant responses.

By addressing these challenges and integrating cutting-edge AI technologies, the project seeks to revolutionize the virtual assistant landscape, providing users with an unprecedented conversational experience that is both personalized and human-like.

1.4 Objectives

To develop an AI modulated voice companion that can also be used to set reminders for OCD patients.

- **reminder** - For timely administration of tasks.
- **Story teller** -The storyteller feature captivates users with engaging and imaginative tales, providing an enjoyable and entertaining experience.
- **Pronouncer** - The pronunciation feature assists users in correctly pronouncing words, contributing to language fluency and confidence in communication
- **Rapper**- The rapper feature showcases the voice assistant's creative side by generating playful and rhythmic responses in a rap style.
- **Dictionary**- The dictionary feature offers a valuable resource for users to quickly access word meanings, enhancing language understanding and communication.

1.5 Scope

The scope of our project is to develop an advanced AI-modulated voice assistant aiming to create a highly interactive and context-aware conversational experience. Advanced nat-

ural language understanding techniques will be implemented to accurately comprehend user queries, allowing the system to handle a wide range of inputs and multi-turn interactions. Personalization features will be explored to tailor responses based on individual user preferences and historical interactions. The project also aims to achieve human-like responses, enhancing user engagement and satisfaction. Ethical considerations regarding privacy, data handling, and potential biases in responses will be addressed. The final product will be scalable, with multi-language support and user-friendly documentation and interface for easy deployment and use.

Chapter 2

Literature Review

2.1 SARA: A Voice Assistant Using Python.

The research paper titled "SARA: A Voice Assistant Using Python" focuses on the utilization of artificial intelligence technologies in human life, particularly in the context of the Internet of Things (IoT) and social networks. The paper emphasizes the significance of recognizing human natural language as a crucial trend in artificial intelligence, which can lead to enhanced human-machine interaction.

The paper acknowledges the active utilization of artificial intelligence technologies in various aspects of human life, especially with the advent and widespread distribution of the Internet of Things. The recognition of human natural language is identified as one of the most important trends in artificial intelligence. Understanding human language can lead to new and more natural forms of interaction between humans and computers. The paper introduces voice assistants as tools that can be integrated into intelligent systems. Voice assistants have the potential to be utilized in various applications and scenarios. The paper outlines an approach to establish a local voice assistant without relying on cloud services. This approach allows for greater flexibility and expansion of future applications for such devices.

Overall, the research paper appears to explore the development of a voice assistant using Python, particularly in the context of artificial intelligence and natural language processing. It aims to contribute to advancements in human-machine interactions and the integration of voice assistants into smart devices and IoT systems.

2.2 Voice assistant integrated with chat gpt

This paper focuses on creating a virtual assistant for the gaming community by integrating Chat GPT technology with voice assistants. With the widespread adoption of

AI-powered voice assistants and chatbots in various fields, this project aims to leverage AI's potential to elevate the gaming experience. By combining voice recognition, natural language processing, and machine learning, the voice-enabled virtual assistant introduces a novel approach for players to familiarize themselves with game mechanics, settings, and map locations within the play-to-hash shooter, Farcana. Beyond assisting in navigating the game, the assistant prioritizes player account security through advanced client authentication and efficiently categorizes user queries to deliver seamless and personalized responses, ensuring an immersive and tailored gaming journey. This project exemplifies the fusion of AI advancements, voice technology, and gaming, promising an unparalleled gaming experience for players in the digital age.

By adopting the Chat GPT approach and leveraging the capabilities of voice assistants, Farcana sets itself apart as an innovative solution for gamers and the gaming community at large. Embracing the latest AI trends, this project enables players to access critical information and game mechanics with ease, making it an indispensable tool for both beginners and experienced gamers. As the digital society continues to embrace AI-driven technologies, the combination of voice-enabled assistance and advanced gaming features promises to revolutionize the gaming landscape and create a more engaging and immersive gaming environment.

The convergence of AI advancements, voice recognition technology, and gaming represents a groundbreaking development in the digital world. Farcana's voice-enabled virtual assistant showcases the power of AI in enhancing customer experiences and streamlining complex processes. As the project paves the way for AI's integration with gaming, it holds the potential to shape future interactions between players and games, making it a trailblazing endeavor that contributes to the overall progress of AI in the digital society.

2.3 The voice enabled personal assistant for PC using python.

This paper focuses on the development of a voice-enabled personal assistant designed to enhance user productivity and convenience. They offer functionalities like making calls, sending messages, taking photos, storing to-do lists, and browsing the internet. By utilizing a virtual assistant, individuals can save considerable time and effort, enabling them to focus on their priorities, whether it's personal or professional. Moreover, the

project recognizes the need for automating routine tasks, as people often waste hours searching for unfamiliar applications in their work environments. Therefore, implementing a voice-enabled personal assistant can significantly streamline this process, allowing users to provide voice commands while the assistant takes care of the rest, leading to greater efficiency and time-saving benefits.

The proposed methodology involves leveraging Python’s Speech Recognition library, equipped with various in-built functions to understand user commands, and the Text-to-Speech library for generating voice responses. When the assistant captures voice commands, underlying algorithms convert them into text, and relevant actions are performed based on the keywords present in the text. The project also harnesses APIs, such as Wolfram Alpha, for performing calculations and extracting news from web sources. Additionally, Python libraries like pyautogui and psutil contribute to functionalities like capturing screenshots and checking battery status. Emphasizing versatility, the project is implemented using Python 3.8.3 and Microsoft Visual Studio Code as the integrated development environment (IDE). The voice-enabled personal assistant is user-friendly, efficient, and proficient in handling queries from people with different voices, providing the flexibility of natural language commands.

The system architecture consists of three key modules: the assistant’s acceptance of voice input from users, analyzing the input to identify the respective intent and function, and delivering results through voice commands. Upon receiving voice input, the assistant converts analog voice to digital text for processing. If there are any conversion challenges, the assistant prompts the user for the input again. Once converted, it proceeds to analyze the input, mapping it to the appropriate function. Subsequently, the assistant delivers the output through voice commands, ensuring a seamless and effective user experience. Overall, this voice-enabled personal assistant is poised to revolutionize how users interact with their devices, simplifying tasks and offering an efficient and user-friendly voice interface.

2.4 Comparison

Comparing the projects, the first paper focuses on the utilization of artificial intelligence technologies in human life, particularly in the context of the Internet of Things (IoT) and

social networks. By integrating an AI assistant with Chatbot GPT, you can leverage the strengths of both systems. The AI assistant provides personalized assistance, task automation, and user-specific context, while Chatbot GPT offers high-quality language generation and creative responses. The second project targets the gaming community with specialized game-related assistance while the third project is a general-purpose voice assistant aimed at everyday tasks. Our project seems to be a more advanced and versatile voice assistant that leverages the capabilities of ChatGPT to handle complex queries and provide more sophisticated interactions. The third paper focuses on the development of a voice-enabled personal assistant designed to enhance user productivity and convenience. Our project could save responses as text files or audio files. The capability to save responses as text and audio files further augments the user experience, making the system a valuable asset in modern-day living. Overall, this literature review provides a comprehensive examination of the integration of Chatbot GPT into a voice-enabled personal assistant, highlighting the system's potential to transform user interactions and productivity. The synthesis of existing research and user feedback serves as a foundation for further advancements and improvements in the field of voice-enabled AI assistants.

Chapter 3

System Analysis

3.1 Expected System Requirements

The system of user which is a smart phone is expected to have the following features:

- Android platform with a version above 4.
- Requirement of Internet connection for speech recognition.
- A storage space of approximate 100 MB for the app.
- A minimum Ram size of 2GB is required in the device.
- SIM card with active connection for working of SOS feature.

3.2 Feasibility Analysis

3.2.1 Technical Feasibility

For virtual assistant, user must have microphone to convey their message and a speaker to listen when system speaks. These are very cheap now adays and everyone generally, possess them. Besides, system needs steady internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.

3.2.2 Operational Feasibility

The operations are built in a simple and easy to use manner. Kids who still don't know to write can read out problems for system and get answers. Installation of the app is the only prerequisite operation to be done.

3.2.3 Economic Feasibility

For this project, the main cost is documentation cost. User also would have to pay for microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, it won't cost too much.

3.3 Hardware Requirements

The following are the system requirements to develop the Julia App.

- Android compatible device
- Hard Disk: Minimum 750MB
- RAM: 6-8GB RAM

3.4 Software Requirements

The following are the softwares used in the development of the app.

Operating System: Windows or Linux

3.4.1 Visual Studio Code (Python(ver.3.11))

Visual Studio Code (VS Code) is a lightweight, fast, and customizable source-code editor developed by Microsoft. It offers cross-platform support, intelligent code editing, integrated terminal, version control integration, and a vast extension library. Its user-friendly interface and active community make it a popular choice for developers across various programming languages and platforms.

Visual Studio Code (VS Code) offers a wide array of powerful tools to enhance the coding experience. It includes IntelliSense for intelligent code suggestions, code navigation features, an integrated terminal, and a user-friendly debugger. Additionally, the editor seamlessly integrates with version control systems like Git and provides access to an extensive extensions marketplace for added functionalities.

Chapter 4

Methodology

4.1 Explanation of each block

4.1.1 User's Voice Command

The User's Voice Command component serves as the entry point for user interactions with the voice assistant. Users can interact with the system by speaking voice commands or queries, which can encompass a wide range of instructions, questions, or requests for information. This component should be designed to efficiently capture and process the user's voice input, ensuring accuracy and clarity in understanding their commands

4.1.2 Speech to Text API

The Speech-to-Text (STT) API plays a vital role in converting the user's voice command into text format. It is essential to carefully select a reliable and accurate STT service provider to ensure precise transcription of spoken words. Proper pre-processing of the text is necessary to clean and normalize the input, removing any background noise or artifacts that may affect the subsequent processing steps. Additionally, thorough error handling should be implemented to handle cases where the transcription may not be accurate or if the STT API encounters any disruptions or latency issues.

4.1.3 ChatGPT API:

The ChatGPT API acts as the brain of the voice assistant, responsible for understanding the natural language input from the user and generating human-like text responses. The choice of language model, such as GPT-3.5 or any other suitable model, is crucial to ensure that the voice assistant comprehends a wide variety of user queries and provides contextually relevant and coherent responses. The integration of the ChatGPT API with

the voice assistant system should be seamless and optimized for efficient processing to ensure quick and accurate responses.

4.1.4 Voice Assistant Output (Textual):

Once the ChatGPT API processes the user's text input, it generates textual responses or actions. The Voice Assistant Output, in the form of text, will be communicated to the user. The module should ensure that the responses are contextually relevant, coherent, and presented in a natural language format to create a seamless and human-like conversational experience.

4.2 Methodology for Implementing the ChatGPT API Voice Assistant:

To effectively implement the ChatGPT API Voice Assistant, the following points should be considered:

4.2.1 Model Selection:

Carefully choose a language model that aligns with the voice assistant's intended use case. Factors to consider include the model's language understanding capabilities, coherence in generating responses, and responsiveness.

4.2.2 API Integration:

Seamlessly integrate the selected language model (e.g., ChatGPT API) with the voice assistant system, ensuring smooth communication between the components. This integration should be optimized for real-time processing to deliver prompt responses.

4.2.3 Error Handling:

Implement robust error handling mechanisms to gracefully deal with any unexpected disruptions or errors that may occur during the interaction with the ChatGPT API. This will help maintain a smooth user experience and avoid potential glitches in the system.

4.2.4 Personalization and Context Retention:

Implement a context retention mechanism to ensure that the voice assistant maintains context across interactions and provides personalized responses. This can enhance the user experience by allowing the voice assistant to remember previous queries and responses, making conversations feel more natural and seamless.

4.2.5 User Interface:

Design an intuitive and user-friendly interface that allows easy interaction with the voice assistant. Consider various interaction modes, such as voice-based commands, touch, or gestures, to cater to different user preferences and accessibility needs.

4.2.6 Testing and Optimization:

Thoroughly test the voice assistant system with a diverse range of user queries to ensure its accuracy, efficiency, and usability. Continuously optimize the system to enhance its performance and user experience, making it even more user-centric and responsive.

4.2.7 Security and Privacy:

Prioritize user data privacy and implement robust security measures to protect sensitive information. Ensure compliance with data protection regulations and establish a transparent privacy policy for users.

Chapter 4: Methodology

Block Diagram:

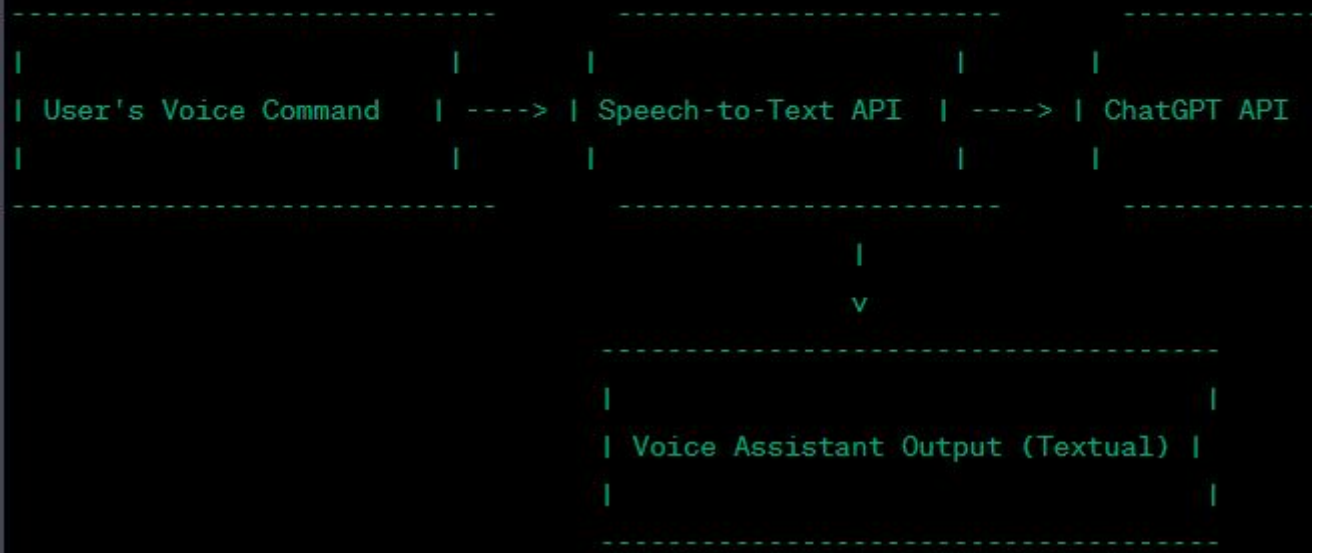


Figure 4.1: Block Diagram

Chapter 5

System Design

5.1 Architecture Diagram

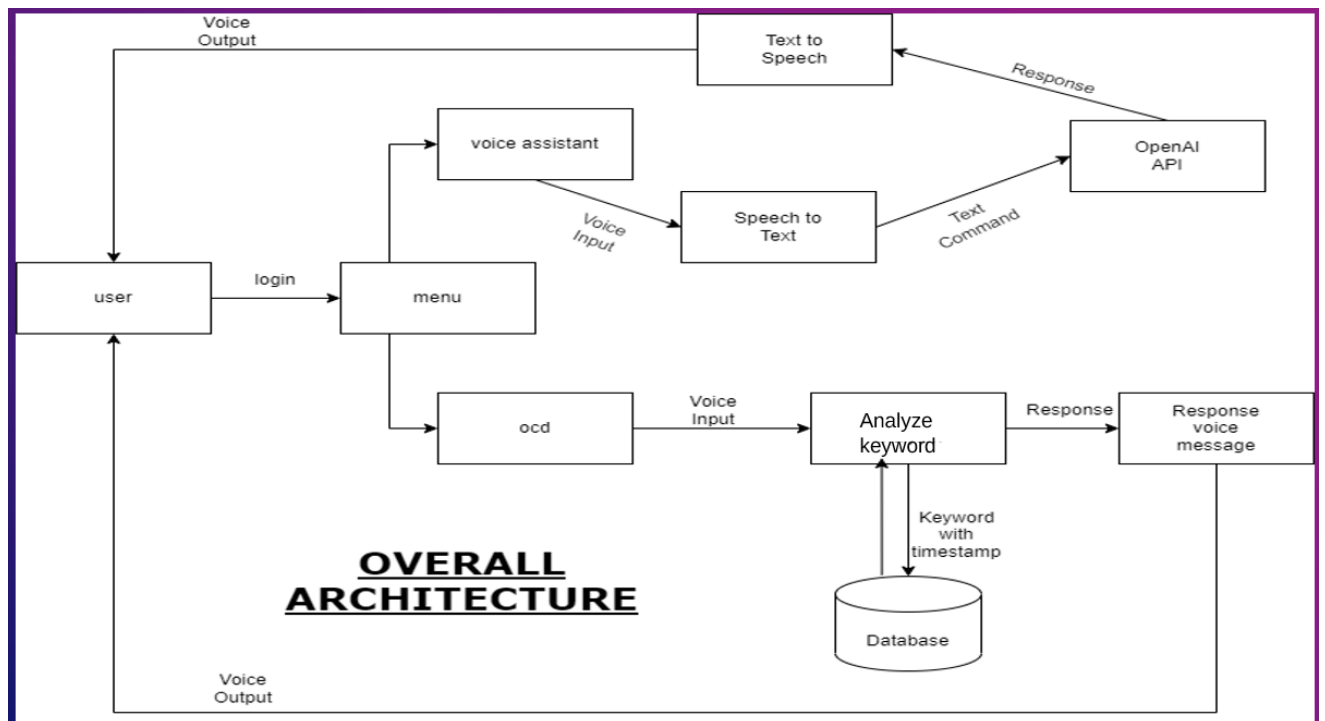


Figure 5.1: Architecture diagram

5.2 Sequence diagram

5.2.1 Login

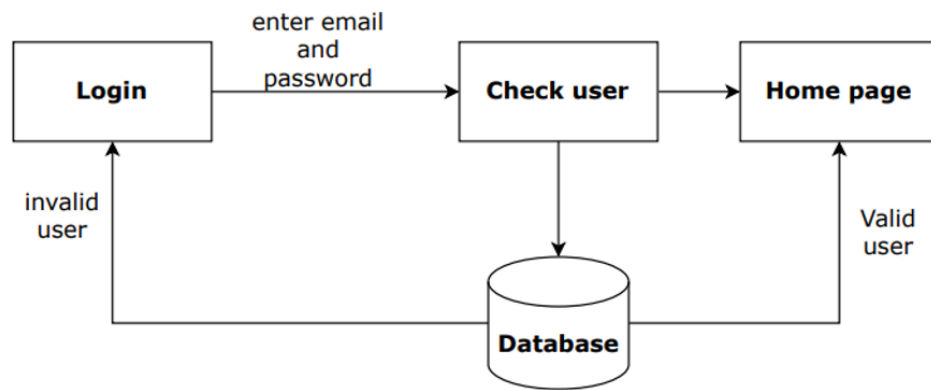


Figure 5.2: Login

5.2.2 Speech to Text

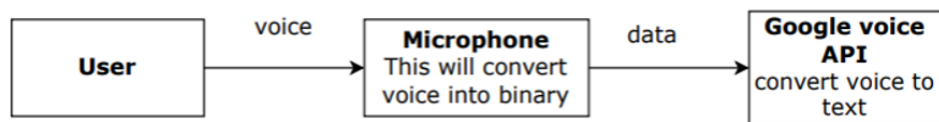


Figure 5.3: Speech to Text

5.2.3 OpenAPI connection

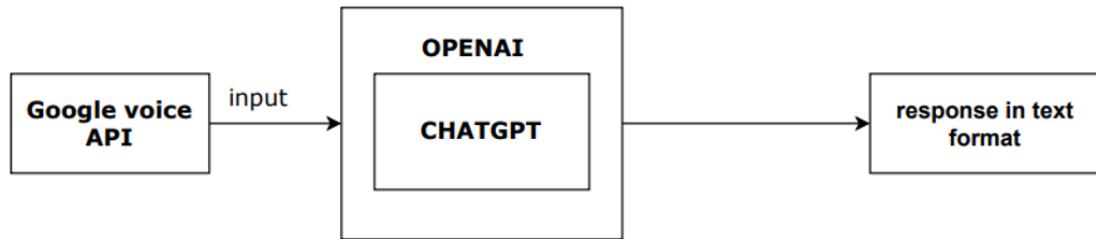


Figure 5.4: OpenAPI connection

5.2.4 Text to Voice



Figure 5.5: Text to Voice

5.2.5 Remainder

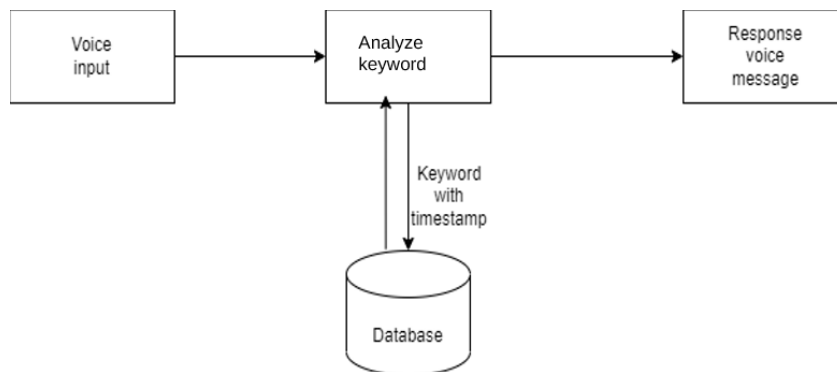


Figure 5.6: Task remainder

Chapter 6

System Implementation

6.1 Login

The login module is a critical component that facilitates user authentication and personalized interactions. This module enables new users to register and create accounts, providing their email address or phone number and choosing a password. User credentials are securely stored in a database during the registration process. To access the system's features, users must authenticate themselves by logging in with their registered credentials. The login module verifies the user's entered username/email and password against the stored records. Robust authentication mechanisms, such as hashing and salting passwords, enhance security and protect against unauthorized access. Account recovery and security features are also included in the login module. Users can reset their passwords via email or phone verification if they forget their login credentials.

Once logged in, the system can personalize the user's experience based on their account information and previous interactions. The AI-modulated voice assistant leverages user preferences, historical queries, and personalized settings to tailor responses and recommendations, enhancing user satisfaction and engagement.

Data privacy and compliance are paramount concerns for the login module. User data, such as passwords and personal details, is encrypted and stored securely to safeguard user information. The system adheres to relevant data protection regulations and obtains explicit user consent for data processing.

6.2 Speech to Text

The Speech-to-Text module is a crucial component responsible for converting user speech into text, enabling seamless interactions between users and the virtual assistant. Leveraging advanced speech recognition algorithms, this module ensures accurate and efficient

speech-to-text conversion, paving the way for natural language understanding and personalized responses.

The process begins when the user activates the voice assistant by issuing a voice command or prompt. The Speech-to-Text module captures the audio input through the device’s microphone and processes it using sophisticated speech recognition models.

Once the audio input is processed, the module generates a corresponding text transcript, which is then passed to the ChatGPT language model for response generation. The seamless integration between the Speech-to-Text and ChatGPT modules ensures a smooth transition from voice-based input to text-based responses, creating a dynamic and interactive conversational experience for users. Throughout the development of the Speech-to-Text module, special emphasis is placed on real-time processing and low-latency performance to ensure a responsive and natural user experience.

By accurately converting user speech into text, this module unlocks the potential for seamless and intuitive interactions, enhancing the overall usability and accessibility of the virtual assistant. The robustness, efficiency, and privacy considerations of the Speech-to-Text module contribute to the project’s success in delivering a cutting-edge and user-centric AI-driven voice assistant.

6.3 OpenAPI connection

The OpenAI Connection module is responsible for establishing a seamless connection to the OpenAI platform and accessing the powerful ChatGPT language model. This module enables our voice assistant to interact with ChatGPT, harnessing its state-of-the-art language modeling capabilities to generate contextually relevant and coherent responses to user queries. The connection is established using secure API calls, ensuring data privacy and confidentiality during the interaction.

Upon receiving the user’s text input from the Speech-to-Text module, the OpenAI Connection module sends the query to the ChatGPT API, initiating a conversation with the language model. The connection effectively facilitates a back-and-forth exchange of messages between the voice assistant and ChatGPT, allowing for dynamic and responsive interactions.

To enhance the user experience, the module is optimized for real-time processing,

minimizing latency and ensuring that responses are generated swiftly, providing an instant and natural conversational flow. The efficient connection with OpenAI’s infrastructure guarantees reliable and high-performance access to ChatGPT, enabling users to enjoy the full capabilities of this cutting-edge language model.

Furthermore, the OpenAI Connection module incorporates error handling mechanisms to gracefully manage any unexpected disruptions during communication with the language model. In case of any issues with the API, the module employs fallback strategies to maintain a seamless user experience and prevent interruptions in the conversation flow.

In conclusion, the OpenAI Connection module plays a crucial role in our project, facilitating the seamless integration of our AI-driven voice assistant with the powerful ChatGPT language model. This module establishes a robust and secure connection to OpenAI’s platform, enabling dynamic and engaging interactions with ChatGPT and delivering contextually aware and personalized responses to users. The efficient and secure connection with OpenAI ensures a cutting-edge and user-centric AI-driven voice assistant experience, marking a significant milestone in revolutionizing human-computer interactions.

6.4 Text to voice

The Text-to-Voice module empowers the voice assistant to convert text-based responses from ChatGPT into natural and intelligible speech for users. Leveraging advanced Text-to-Speech (TTS) technology, this module seamlessly transforms the generated text into a human-like and expressive voice, enhancing the user’s conversational experience.

Upon receiving the text response from ChatGPT, the Text-to-Voice module processes the text through sophisticated TTS algorithms. These algorithms use deep learning techniques, such as neural networks and WaveNet, to synthesize speech that closely resembles human speech patterns and intonation. This ensures that the voice assistant’s responses sound natural and engaging, enabling users to interact with the virtual assistant as if communicating with another person.

To ensure a seamless and efficient experience, the module is optimized for real-time processing, minimizing any time lag between generating the text response and producing the corresponding speech. This optimization ensures that users receive prompt and

dynamic responses, making the conversation flow feel natural and engaging.

Moreover, the Text-to-Voice module is designed to be resource-efficient, minimizing the computational resources required for speech synthesis. This allows the voice assistant to function efficiently even on devices with lower processing capabilities, ensuring a wide range of users can access the app.

In conclusion, the Text-to-Voice module plays a pivotal role in enhancing the user experience of our AI-driven voice assistant. By seamlessly converting text responses from ChatGPT into expressive and natural speech, this module creates a lifelike and engaging conversational interaction. The customization options, real-time processing, and resource efficiency further contribute to the project's success in delivering a cutting-edge and user-centric voice assistant.

6.5 Remainder

The program module is designed to interact with users through a user-friendly interface, allowing them to input task names using voice input. Upon receiving a new task, the program will capture the current timestamp to record the time of task creation. To achieve this functionality, the program will utilize data structures, such as arrays or lists, to store task objects, where each object will contain the task name and its corresponding timestamp. Additionally, a Boolean flag will be associated with each task to represent its completion status (done or not done).

The program will offer two main functionalities: "Add Task" and "Check Task." The "Add Task" function will prompt users to input the task name, and upon confirmation, it will create a new task object with the timestamp and completion status defaulting to "not done." The task object will then be added to the list of tasks.

For the "Check Task" function, the program will prompt users to enter the task name they wish to inquire about. The program will search the list of tasks for a match, and if found, it will return the task's completion status along with the timestamp when it was added. If the task name is not found in the list, the program will inform the user that the task does not exist.

To ensure a smooth user experience, the program will include error handling to deal with potential input mistakes or edge cases. It will provide appropriate feedback to users

in case of incorrect input, guiding them to retry or input valid task names. Additionally, the program will allow users to update task completion status if they have completed a specific task.

Overall, this methodology focuses on creating an efficient and user-friendly task management program that allows users to add tasks with timestamps and check the completion status of specific tasks. By utilizing appropriate data structures and error handling techniques, the program will deliver a reliable and intuitive user experience for efficient task tracking and management.

Chapter 7

Testing

In our project, testing plays a crucial role in ensuring that it meets the highest standards of reliability, functionality, and performance. Through a comprehensive and meticulous testing process, we verify that all components and features of the voice assistant work seamlessly together to deliver an exceptional user experience.

7.1 Unit Testing:

To ensure the accuracy and efficiency of individual components, we conduct unit testing. This involves testing each module in isolation to verify that they function as intended and produce correct outputs. For instance, the Speech-to-Text module is thoroughly tested to ensure that it accurately converts user speech into text, while the Text-to-Voice module is validated to generate natural and expressive speech responses.

7.2 Integration Testing:

Integration testing is performed to assess the smooth interaction and data flow between different modules within the voice assistant. We rigorously test how the Speech-to-Text module communicates with the ChatGPT integration, ensuring that information exchange is seamless and error-free.

7.3 Functional Testing:

Functional testing is a pivotal part of the testing process, where we validate the voice assistant's functionalities against predefined requirements and use cases. We thoroughly examine each feature, including the voice assistant, task reminder, storyteller, pronunciation assistance, and dictionary, to ensure they all perform as expected.

7.4 User Acceptance Testing (UAT):

To ensure that the voice assistant meets user expectations, we involve real users in user acceptance testing. Users validate whether the voice assistant fulfills their requirements and provides a satisfactory experience.

Through these meticulous testing processes, we guarantee that the "Advanced AI-Modulated Voice Assistant Integrated with ChatGPT" is a robust and reliable system, delivering an exceptional and seamless user experience. Testing is an ongoing process that continues throughout the development lifecycle, allowing us to address any issues promptly and refine the voice assistant for optimal performance and user satisfaction.

Chapter 8

Results

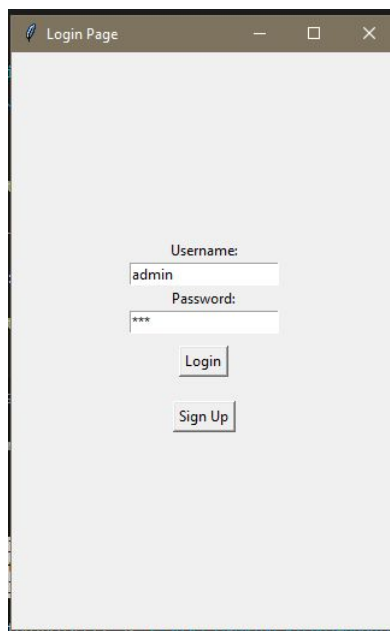


Figure 8.1: Login page



Figure 8.2: Main menu

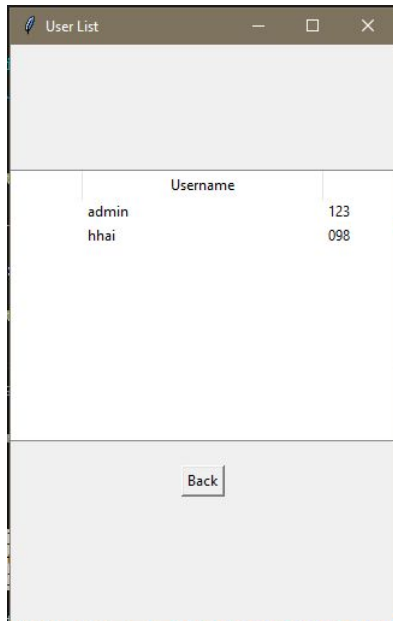


Figure 8.3: Users Page

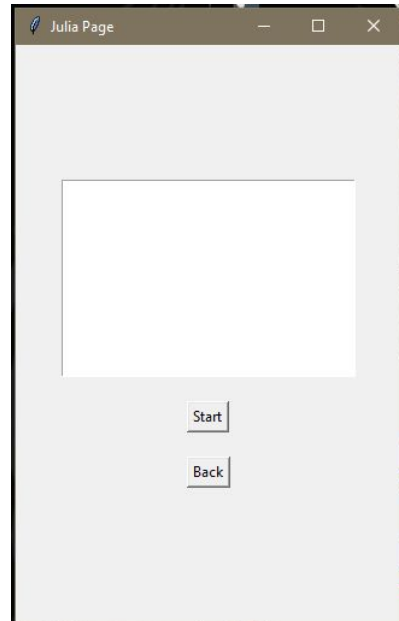


Figure 8.4: AI Assistant interface



Figure 8.5: First input: write a letter

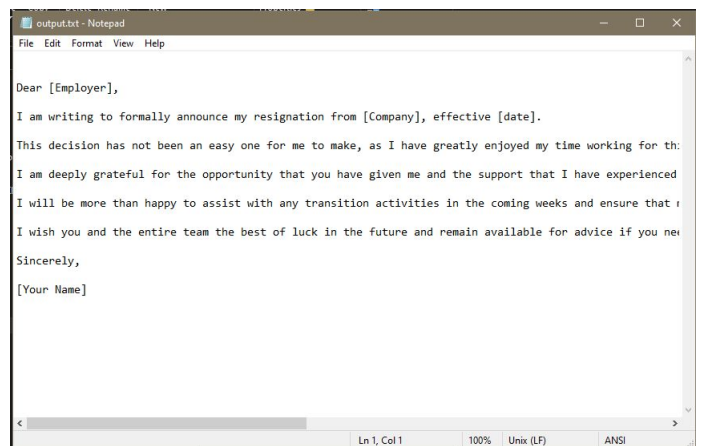


Figure 8.6: Output: writing letter

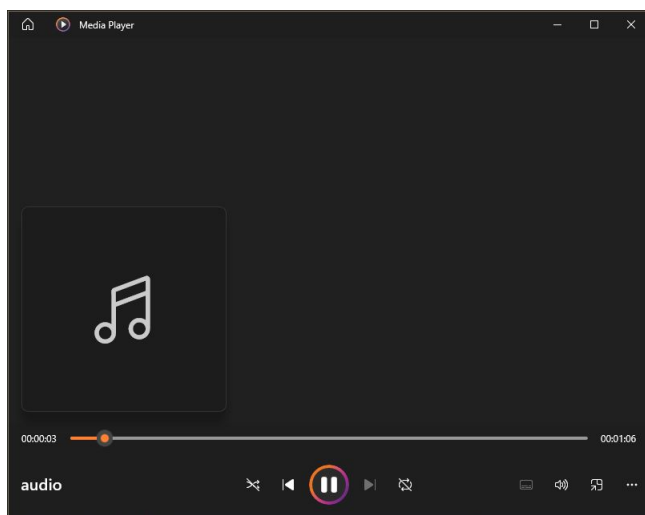


Figure 8.7: Response saved as audio file

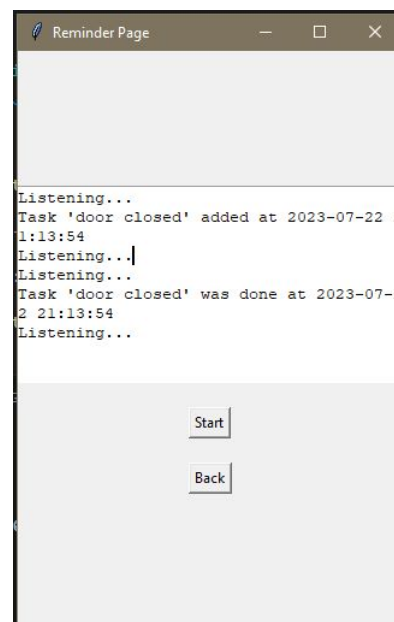


Figure 8.8: OCD assistance

Chapter 9

Risks and Challenges

1. Currently, the language support is limited to English only.
2. There might be a time lag in generating responses, affecting real-time interactions.
3. The system requires a high-speed internet connection to access the language model and provide prompt responses.

Chapter 10

Conclusion

We have developed an Advanced AI-Modulated Voice Assistant Integrated with ChatGPT. This app consists of the following features: Voice assistant, Task remainder, Storyteller, Dictionary, Pronunciation. All of these features have been verified to be working as intended. With our app, users can experience the full potential of AI-driven technology, embracing a personalized and dynamic user experience with the voice assistant.

Looking forward, our project's future scope is promising, with the potential for further advancements in multimodal integration, cross-lingual support, real-time processing, and specialized domain applications. Our journey towards refining and expanding the capabilities of our voice assistant is a testament to our dedication to continuous improvement and innovation.

References

- [1] Chinchane, Ayush, Aryan Bhushan, Ayush Helonde, and Kiran Bidua. "SARA: A Voice Assistant Using Python." *International Journal for Research in Applied Science and Engineering Technology* 10, no. 6: 3567-3582.
- [2] Shafeeg, Abdulla, Ilman Shazhaev, Dimitry Mihaylov, Arbi Tularov, and Islam Shazhaev. "Voice assistant integrated with chat gpt." *Indonesian Journal of Computer Science* 12, no. 1 (2023).
- [3] 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 (2021): 162-165.
- [4] Dhanraj, Vishal Kumar, and Semal Mahajan Lokeshkriplani. "Research Paper on Desktop Voice Assistant." *International Journal of Research in Engineering and Science (IJRES)* 10, no. 2 (2022): 15-20.
- [5] Kumar, C. Raju. "Virtual assistant using artificial intelligence and Python." *J. Emerg. Technol. Innov. Res* 7, no. 3 (2020): 1116-1119
- [6] Shende, Deepak, Ria Umahiya, Monika Raghorte, Aishwarya Bhisikar, and Anup Bhange. "AI based voice assistant using python." *Journal of Emerging Technologies and Innovative Research (JETIR)* 6, no. 2 (2019): 506-509

Appendix A: Base Paper



iJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VI **Month of publication:** June 2022

DOI: <https://doi.org/10.22214/ijraset.2022.44517>

www.ijraset.com

Call: ☎ 08813907089

E-mail ID: ijraset@gmail.com

SARA: A Voice Assistant Using Python

Ayush Chinchane¹, Aryan Bhushan², Ayush Helonde³, Prof. Kiran Bidua⁴

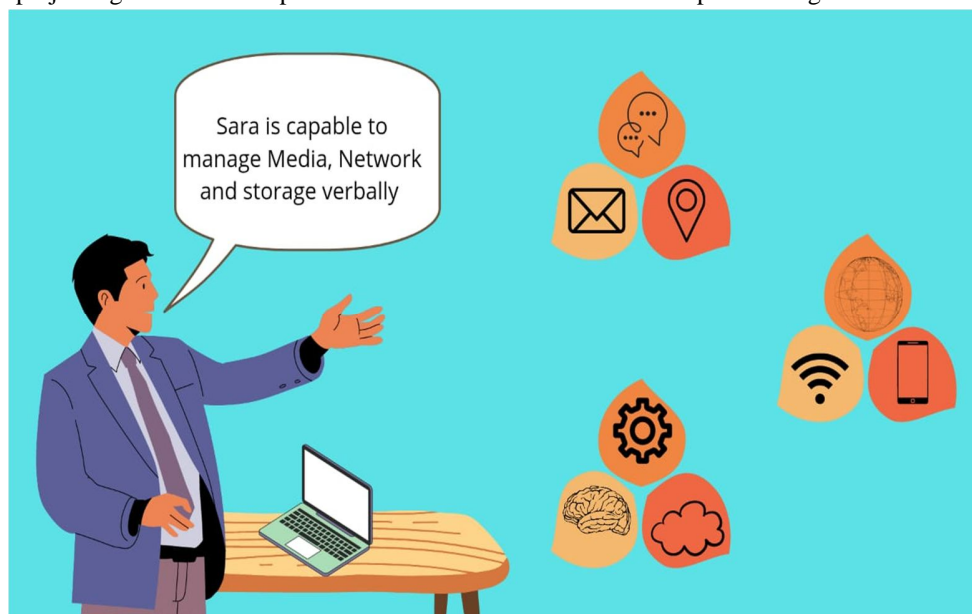
Abstract: Artificial intelligence technologies are starting to be actively utilized in human life, thanks to the Internet of Things' debut and widespread distribution. Autonomous gadgets are growing more intelligent in their interactions with humans and with each other. New capabilities led to the development of various solutions for integrating smart devices into the Internet of Things Social Networks. The technique of recognizing a human's natural language is one of the most important trends in artificial intelligence. New insights into this area could lead to new forms of natural human-machine interaction, in which the computer learns to understand and engage with human language.

Voice assistant is one of these tools, and it can be integrated into a variety of different intelligent systems. The basics of voice assistant operation are outlined in this paper, as well as the major flaws and limitations. The approach for establishing a local voice assistant without needing cloud services is explained, allowing future applications of such devices to be considerably expanded.

Keywords: Artificial Intelligence, Machine Learning, Natural Language Processing, Voice detection, Voice recognition

I. INTRODUCTION

Nowadays, humans rely on other humans for help or services. The world's digitalization ensured that human reliance on the system could be switched to the system, allowing for far more efficient and reliable employment, as well as a device that could take care of their daily needs. Computers, cell phones, laptop computers, and other electronic devices have become an indispensable part of our daily lives. They can perform simple calculations as well as complex programs, reducing monotonous work and personnel waste. To solve problems quickly, Virtual Personal Assistants have practically become a must-have feature in all electronic devices. Virtual assistance can help the user in a variety of ways. Speech recognition is a relatively new addition to the virtual world. However, despite being reasonably effective, it is not particularly useful and, as a result of the high rate of error, is not used by the user. Despite the fact that the future virtual assistant has an error rate of about 5%, it is not yet ready to become a routine part of the user's life. As a result, the project's goal is to develop a virtual assistant with low error rate speech recognition.



We developed a voice assistant that allows users to accomplish any task on the system without having to use a keyboard, decreasing the number of input devices.



The elderly, the visually and physically handicapped, children, and others benefit from virtual assistants since engaging with machines is no longer a challenge. Even blind people who can't see the computer can communicate with it simply by speaking to it. Some of the basic tasks that a voice assistant can assist you with are listed below.

- 1) Reading Newspaper
- 2) Getting updates on mail
- 3) Search on the web
- 4) Play music or video
- 5) Setting a reminder and alarm
- 6) Run any program or application
- 7) Getting weather updates

These are some of the examples, we can do many more things according to our requirements.

II. COMPREHENSIVE EXISTING WORK SURVEY

Works	Name	Algorithms/Techniques used	Type	Research
[1]	A Voice Based Assistant Using Google Dialogflow and Machine Learning	Artificial Intelligence, Natural Language Understanding, IBM Watson, Google Dialogflow, Speech Recognition	Personal Voice Assistant	In this project, the application, ERAA, developed with the help of Google Dialog Flow, is able to perform various tasks like accessing the other applications like WhatsApp, Instagram, and Gmail that are installed on the device. It is user-friendly and was developed with the help of Flutter, which provided ease in accessing the application. With the help of graphics packages in Flutter, they were able to develop an attractive user interface. It is able to perform the basic features as required in an ideal Personal Assistant.
[2]	Desktop Voice Assistant Using Natural Language Processing (NLP)	Speech Recognition, Python Backend, System Calls, Google-Text-To-Speech	Desktop Voice Assistant	In this study, they have developed a voice assistant that can perform any kind of task in exchange for commands given by the users without any error. They have added more features like listening to the user's voice only and not being activated by environmental noise.
[3]	Desktop Assistant AI Using Python	Desktop Assistant, Python, Machine Learning, Text to Speech, Speech to Text, Language Processing, Voice Recognition, Artificial Intelligence, Internet of Things (IoT), Pyttsx3, Speech Recognition, SQLite	Desktop Voice Assistant	They discussed a Python-based voice-activated personal assistant in this paper. This assistant currently works online and performs basic tasks like weather updates, streaming music, searching Wikipedia, opening desktop applications, etc. The functionality of the current system is limited to working online only.



[4]	JARVIS: A PC Voice Assistant	Python[pyttsx] and gTTS[Google Text to Speech]	PC Voice Assistant	This voice assistant has automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, translating words from one language to another language, accessing youtube videos, sending mail through voice, and solving computational queries.
[5]	The Voice Enabled Personal Assistant for Pc using Python	Python, Quepy, Pyttsx3, Speech Recognition, SQLite	Personal Voice Assistant	This paper presents a comprehensive overview of the design and development of a voice-enabled personal assistant for the PC using the Python programming language. This voice-enabled personal assistant, in today's lifestyle, will be more effective in saving time compared to the previous days. Furthermore, there are many things that this assistant is capable of doing, like turning our PC off, restarting it, or reciting the latest news, with just one voice command.
[6]	Smart Voice Based Virtual Personal Assistants with Artificial Intelligence	Python, Text-to-Speech, Speech-to-Text, Voice Recognition	Virtual Assistant	In this paper, the design and implementation of an Intelligent Personal Voice Assistant are described. The project is built using available open-source software modules with visual studio code community backing, which can accommodate any updates in the future.
[7]	Voice Assistant Using Python	Desktop Assistant, Python, Text to Speech, Virtual Assistant, Voice Recognition	Voice Assistant	This paper discusses a voice assistant developed using Python. This assistant currently works as an application and performs basic tasks like weather updates, streaming music, searching Wikipedia, opening desktop applications, etc.
[8]	AI Based Voice Assistant Using Python	Speech Recognition, Python, Speech-to-Text, Text-to-Speech	Voice Assistant	In this paper, they have developed a voice assistant using Python. The project is built using open-source software modules with the PyCharm community. This project can be further improved by implementing the voice command in Google search queries.
[9]	Personal Assistant with Voice Recognition Intelligence	Google Voice Search, Voice Pattern Detection, Keyword Learning	Personal Voice Assistant	This paper focuses on "PARI", which is specially designed to help Native and Blind people who work on their Voice Commands. It also has the capability of recognizing voice commands without an internet connection. It has various functionalities for mobile devices, like network connection and managing various



				applications with just voice commands. It contains key features like Voice Pattern Detection, Keyword Learning, etc.
[10]	INTELLIGENT VOICE ASSISTANT	wake-word, voice assistant, voice recognition, Alexa, API-Application Program Interface, localization	Voice Assistant	This intelligent voice assistant responds to the commands given by the user. To accept the command, first, the voice recognition tool of the system has to be awakened to accept and execute the request. In this, various skills have been created in Hindi and Marathi languages, such as facts, good thoughts, weather, time, nearby hospitals, and a city guide. These skills are created using an Amazon Developer account.
[11]	Desktop Voice Assistant	Speech recognition, Python, and Google text-to-speech	Voice Assistant	Here they have developed a voice assistant which can perform any kind of task in exchange for commands given by the users without any error. They have also added more features to it, like that it will listen to the user's voice only and will not be activated by environmental noise. All the packages required in the Python programming language have been installed and the code was implemented using the VS Code Integrated Development Environment (IDE).
[12]	Vitro: Designing a Voice Assistant for the Scientific Lab Workplace	Design Research, Conversational Agent, Augmented Scientific Workplace	Voice Assistant	In this paper, they have designed a voice assistant and also done research on how the voice assistant can play a major role in a laboratory.
[13]	Firefox Voice: An Open and Extensible Voice Assistant Built Upon the Web	Conversational user interface, CUI, Browser Extension, OpenSource	Open source voice assistant	This paper mainly focuses on "Firefox Voice", a voice assistant which is developed by the Mozilla Foundation and its subsidiary, the Mozilla Corporation. It shows how the voice assistant works and what its features are.
[14]	Voice Assistant Application for the Serbian Language	Voice assistant, continuous speech recognition, Kaldi speech recognition toolkit, Serbian, Android	Voice Assistant	This Voice Assistant is the first mobile phone application developed for the Serbian language, allowing faster and more natural communication between the phone and the user, even under noisy conditions. The results were highly improved by incorporating the noise itself within the acoustic model.
[15]	Home Automation using Arduino and Smart Phone	Nodemcu microcontroller, Google Assistant APP, IFTTT, Adafruit IO	Voice assistant for home	It is a home automation system using the Arduino Uno board and wireless fidelity technology. It accepts commands only through clicks. Home automation through Android



				mobile is designed for physically challenged and disabled people.
[16]	A voice based text mail system for visually impaired	Voice-based, Visually handicapped, Email System	Voice-based Text Mail System	Here they have proposed an android application designed specifically for visually challenged people. It provides a voice-based mailing service where they can read and send mail on their own, without any guidance. The users have to use certain keywords which will perform certain actions, e.g., read, send, compose mail, address book, etc. This email system can be used by a blind person to access mail easily and efficiently.
[17]	Voice Control Human Assistance Robot	Raspberry pi, Assistant, voice recognition, etc	Voice Controlled Robot	The robot developed in this project is able to move in any direction, like the front, back, left, right, according to the voice commands received from the user through a microphone as part of our hardware in this project. There is an autonomous voice command which can instantly make the robot move automatically without hitting any obstacle using an ultrasonic sensor. This device will help users give a uniform look to their lawn with ease. As well, it can also be used for blind or physically challenged people by embedding this system into the wheelchair, which will make an autonomous wheelchair.
[18]	Virtual Assistant in Native Language	Virtual Assistant, Sinhala, Cloud Deployment, Speech recognition, Translation	Virtual Assistant	In this paper, they have discussed the Sinhala language. The language is useful and what major role does it play in today's tech industry? So, around 8 million people all around the world use the Sinhala language as their preferred oral language. Most of them are associated with technological advancements. Most Sinhala people suffer from finding the correct words in English. So, this is a start-up solution for such people to virtually assist in their mother tongue to get work done. This project gives a basis for how virtual assistants work and also gives a basis for what is probably the fastest way to get Sinhala to make an identity in the tech industry.
[19]	Voice Control Device using Raspberry Pi	Virtual Personal Assistant, Natural Language Processing, Query Processing,	Voice-Controlled Device	This paper describes the working of a device based on the implementation of a voice command system as an intelligent personal assistant. This voice-driven device uses the

		Raspberry Pi		Raspberry Pi as its main hardware. A speech-to-text engine is used to convert the voice command to simple text. Query processing is then applied using natural language processing (NLP) to this text to interpret the intended meaning of the command given by the user. After interpreting the intended meaning, text-to-speech conversion is used to give the appropriate output in the form of speech. The services provided by the device depending on the input given, such as weather, telling time, or accessing online applications to listen to music.
[20]	Speech Emotion Recognition using Neural Network and MLP Classifier	MLP-Classifer, MFCC, Model, Neural Networks, Prediction	Speech Emotion Recogniser	Speech Emotion Recognition (SER) is a technique that uses Neural Networks to classify emotions from a given speech. It is based on the fact that the voice often reflects underlying emotion through tone and pitch. Speech Emotion Recognition helps to classify and elicit specific types of emotions. The MLP-Classifer is used to classify the emotions from the given wave signal, which makes the choice of learning rate adaptive. The dataset used will be RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song dataset).
[21]	Voice Recognition based Intelligent Wheelchair and GPS Tracking System	Voice Recognition, GPS module, smartphone application, Firebase, Wi-Fi module, speed control, obstacle detection	Intelligent wheelchair and GPS Tracking System	Here is a development of a voice recognition-based intelligent wheelchair system for physically handicapped people who are unable to drive the wheelchair by hand. So this system works like this: the patient can operate the wheelchair using voice commands and the location of the patient can be tracked using a GPS module in the wheelchair that tracks and sends the information to a smartphone application (app) via Firebase. The Voice Module V3 is used to record a patient's voice and recognize that voice to follow the instructions of the patient.

III. PROBLEMS IDENTIFIED

Voice AI devices, such as Amazon Alexa, Microsoft Cortana, Apple Siri, and Google Assistant, will be the channel through which we communicate with one another and with our software.

For starters, voice AI systems should ideally provide nuanced responses, making us feel as if we're conversing with a fellow human, or something close to it. Conversational AI is currently a command line, not a genuine dialogue, as everyone with a smart speaker or a personal assistant on their phone knows.

We also faced a few problems:

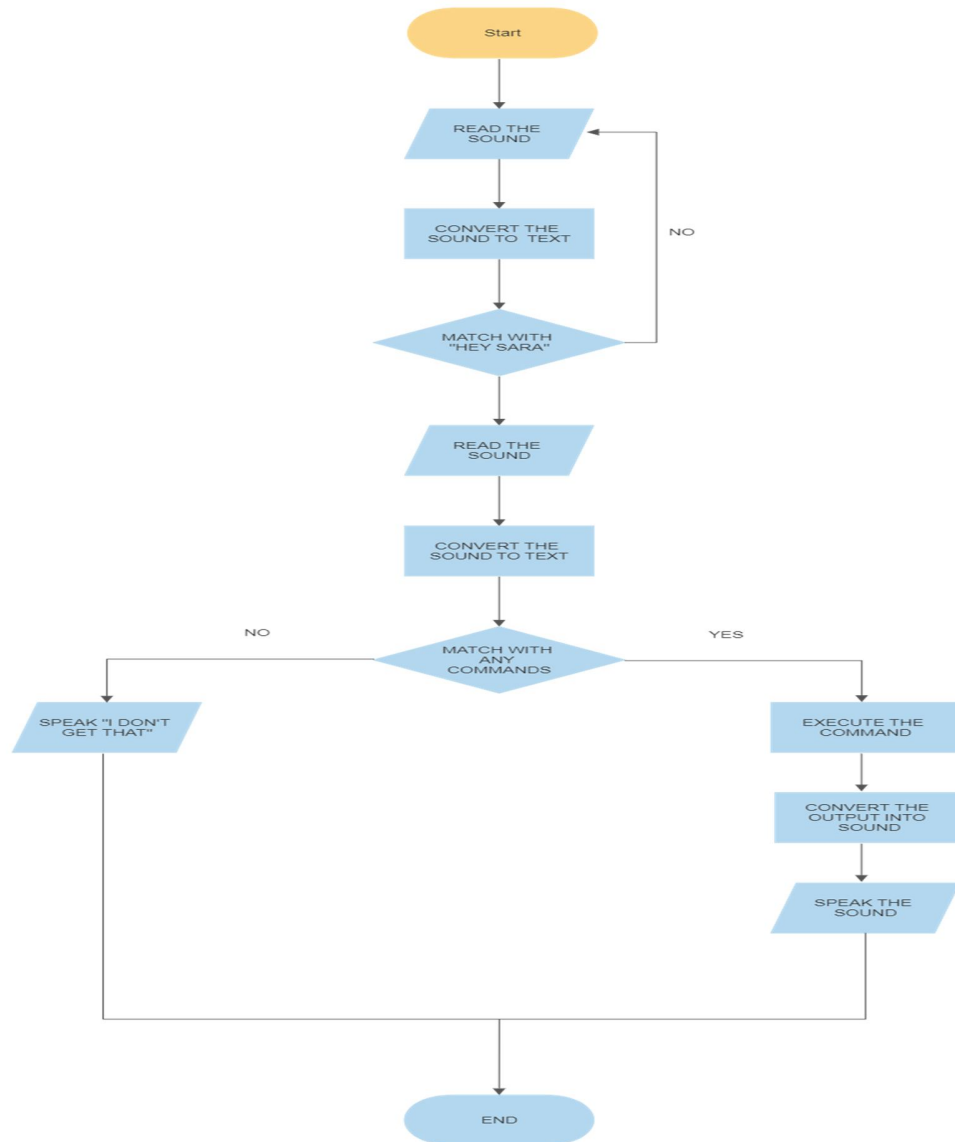
- 1) Regarding the weather forecasting part because we need to pay every time we use it.
- 2) Excluding Anaconda and Jupyter Notebook, we faced many problems while running the program on other software applications. The user interface of Anaconda is easy to work with and it does not give many indentation errors.

The problems that we faced are minor and we are currently working on them. Even though there are some drawbacks in our system, that doesn't mean our system isn't properly working. Leaving the drawbacks, our system is almost capable of doing things that a normal voice assistant can.

IV. PROPOSED APPROACH

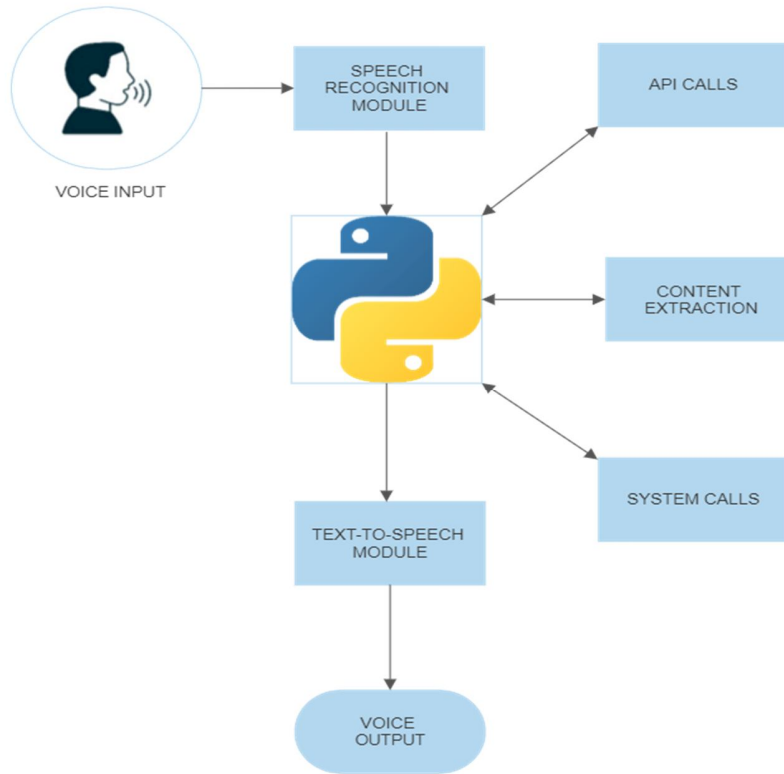
The following features will be included in the proposed system:

- 1) The system will continue to listen for commands, and the length of time it spends listening is adjustable to meet the needs of the user.
- 2) If the system is unable to extract information from the user's input, it will prompt the user to repeat the process until the desired number of times has been reached.
- 3) The system will be voiced by a woman.
- 4) Playing music, sending emails, sending texts, searching Wikipedia, accessing system-installed applications, opening anything in the web browser, and so on are all supported in the present edition.



V. WORKFLOW & METHODOLOGY

The study began with an analysis of the user's auditory commands delivered through the microphone. This can include obtaining any information, accessing the computer's internal data, and so on. This is empirical qualitative research based on reading the material indicated above and putting the instances to the test. Tests are carried out by programming in accordance with books and internet resources, with the stated purpose of discovering best practices and a deeper understanding of Voice Assistant.



Speech Recognition	The system converts speech input to text using Google's online speech recognition system. The voice input Users can obtain texts from the special corpora organized on the computer network server at the information center, which are temporarily stored in the system before being sent to Google cloud for speech recognition. After that, the equivalent text is received and fed into the central processor.
Python Backend	The python backend reads the voice recognition module's output and determines whether the command or speech output is an API Call, Context Extraction, or System Call. The output is then transmitted back to the python backend to provide the user with the desired results.
API Calls	API is an abbreviation for Application Programming Interface. An application programming interface (API) is a software interface that enables two applications to communicate with one another. In other words, we can say that an API serves as a messenger, who can deliver your request to the provider and then return the response to you.

Content Extraction	Context extraction (CE) is the process of extracting structured information from unstructured and/or semi-structured machine-readable documents automatically. In most cases, this activity involves using natural language processing to process human language texts (NLP). Recent developments in multimedia document processing, such as automatic annotation and content extraction from images/audio/video, could be viewed as context extraction TEST RESULTS.
System Calls	A system call is a programmatic method by which a computer program requests a service from the kernel of the operating system on which it is running. This can include hardware-related services, the creation and execution of new processes, and communication with core kernel services like process scheduling. System calls serve as a vital link between a process and the operating system.
Text-to-Speech	The capacity of computers to read text aloud is referred to as text-to-speech (TTS). Written text is converted to a phonemic representation, which is subsequently converted to waveforms that can be generated as sound by a TTS Engine. Third-party publishers offer TTS engines in a variety of languages, dialects, and specialized vocabularies.

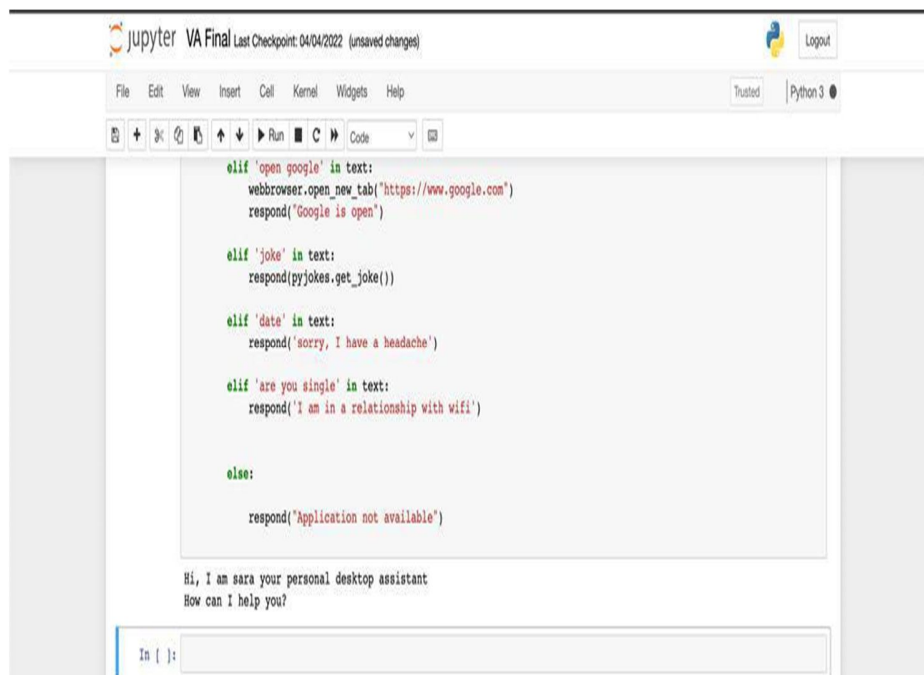
VI. SYSTEM ARCHITECTURE

```
In [23]: import speech_recognition as sr      #convert speech to text
import datetime                             #for fetching date and time
import wikipedia                             #to get data from wikipedia
import webbrowser                           # to brows
from gtts import gTTS                       # google text to speech
import os                                   # to save/open files
import pyjokes                              #for jokes
import playsound                            #to play sound
import pyaudio                              #to play audio
import wolframalpha                         # to calculate strings into formula
from selenium import webdriver              # to control browser operations
import requests                             #used to send all types of HTTP request
```

- 1) *Speech Recognition*: Speech recognition is the ability of a machine to understand what humans are saying. In our project, we're using Python and Google Speech API to develop software that can run devices on command. To recognize voice commands, we must install the Pyaudio Python package. The 'pip install Pyaudio' command is used to install Pyaudio.
- 2) *DateTime*: The DateTime package is used to display Date and Time on our output screen. It comes built-in with Python.
- 3) *Wikipedia*: In our project, we used the Wikipedia module to get more information from Wikipedia or to perform a Wikipedia search. We have used 'pip install wikipedia' to install this Wikipedia module.
- 4) *Webbrowser*: The Webbrowser package is used to perform a web search. It comes built-in with Python.
- 5) *gTTS*: gTTS is abbreviated as Google Text-to-Speech. It converts your audio commands to text. It will basically convert the response from the lookup function that you write to get the answer to the question or command into audio form. This package connects to the Google Translate API.
- 6) *OS*: OS basically stands for Operating System. Python's OS module provides functions for interacting with the operating system. Python's standard utility modules include OS. This module enables the use of operating system-dependent functionality.
- 7) *Pyjokes*: Pyjokes is a tool for collecting jokes from the Internet. We have included Pyjokes in our project because it includes jokes. It's very intriguing. Pyjokes is a one-line joke that adds interest to our project.

- 8) *Playsound*: The Playsound module is a platform-independent module that can play audio files. It is simple to use Playsound on Python. There are no dependencies for this, simply install with pip in your virtual environment and run.
- 9) *Pyaudio*: Pyaudio is a Python binding for PortAudio, a cross-platform audio input/output library. This essentially means that we can use Pyaudio to record and play sound on any platform or operating system, including Windows, Mac, and Linux.
- 10) *WolframAlpha*: WolframAlpha is a Wolfram Research computational knowledge engine and answer engine. It provides direct answers to factual queries by computing the answer from externally sourced data.
- 11) *Selenium*: Selenium is an open-source umbrella project for a variety of browser automation tools and libraries.
- 12) *Requests*: Python's Requests module allows you to send HTTP requests. It's used to send GET and POST requests. It hides the complexities of making requests behind a beautiful, straightforward API.

VII. RESULTS



```
jupyter VA Final Last Checkpoint: 04/04/2022 (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

elif 'open google' in text:
    webbrowser.open_new_tab("https://www.google.com")
    respond('Google is open')

elif 'joke' in text:
    respond(pyjokes.get_joke())

elif 'date' in text:
    respond('sorry, I have a headache')

elif 'are you single' in text:
    respond('I am in a relationship with wifi')

else:
    respond('Application not available')

Hi, I am sara your personal desktop assistant
How can I help you?

In [ ]:
```

This part of the research paper is a brief description of the output of our project. In our project, we chose Python as the preferred programming language. We primarily worked on AI and machine learning. We focused on tasks performed by the voice assistant. The main reason for using Python is that it was easy to deploy in the Jupyter notebook, we have compared other deploying engines and this was comfortable and easy to use.

The output that is shown in the figure is the first thing that our voice assistant responds with .i.e.

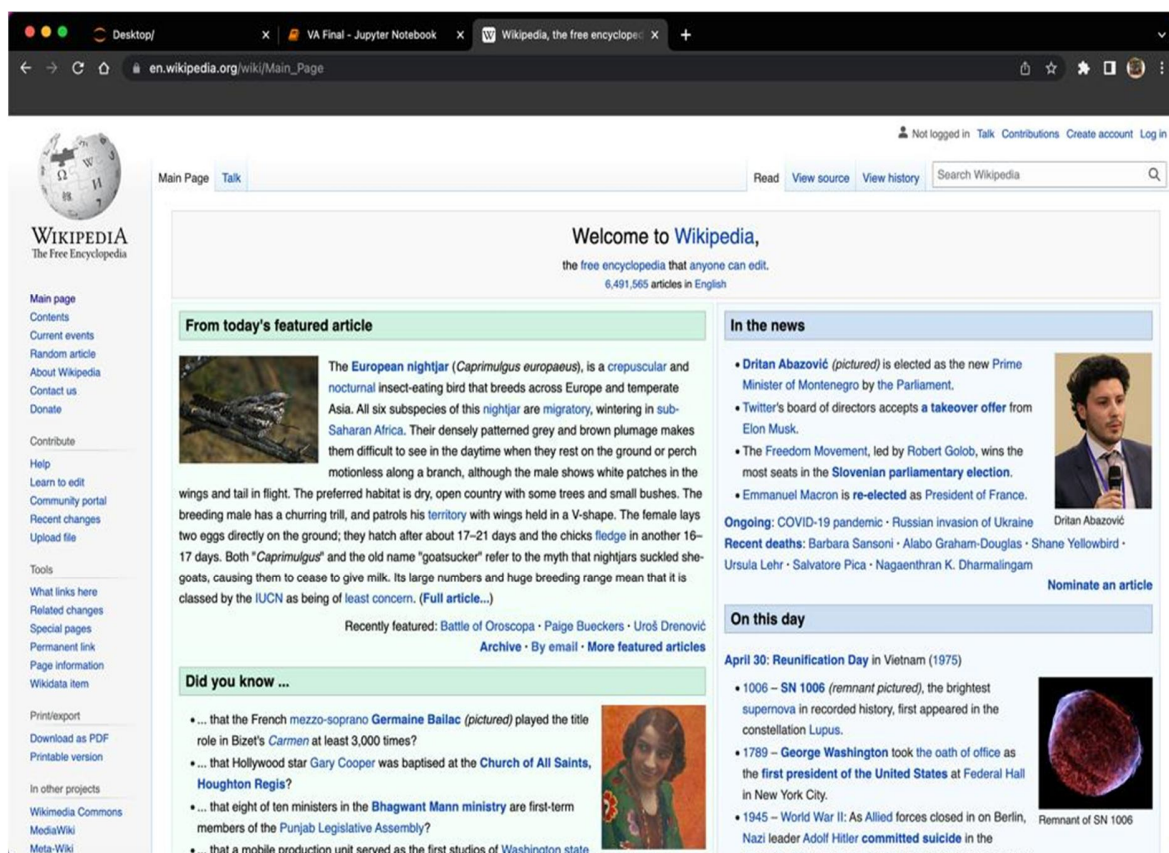
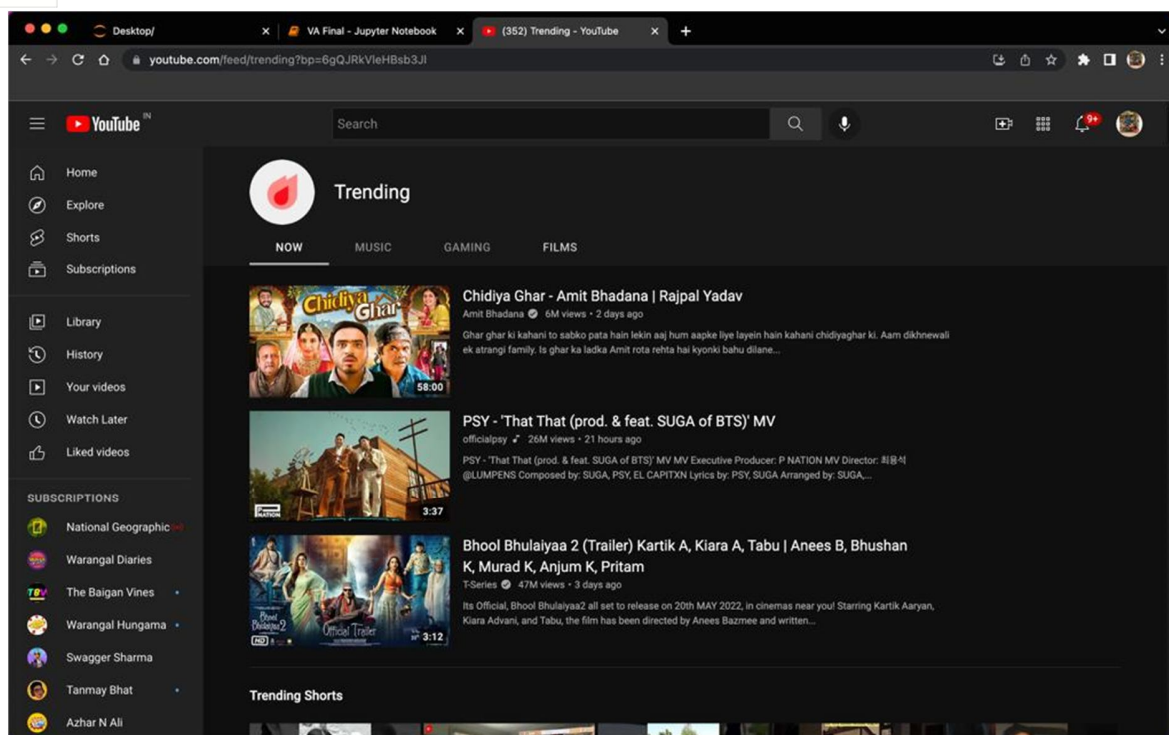
“Hi, I am sara your personal voice assistant”

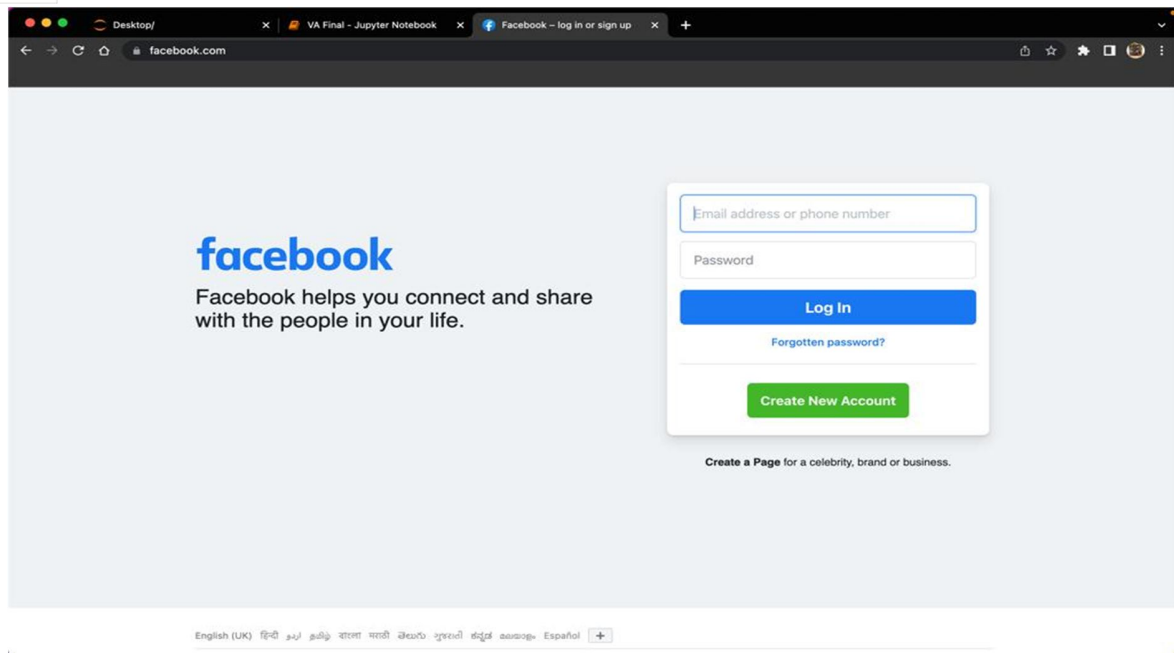
“How may I help you?”

Following this command, the user needs to respond in a particular way that the voice assistant may respond accordingly and give the required output. We have used different algorithms for each command, just like we have used a command that can open Google Chrome, Youtube, Wikipedia, Facebook, etc.

There are a few screenshots of the output that our voice assistant gives on executing the following commands:

- 1) “Open Youtube”
- 2) “Open Wikipedia”
- 3) “Open Facebook”





VIII. COMPARISON BETWEEN OUR PROJECT AND EXISTING PROJECTS

Works	Name	Research	Comparison
[2]	Desktop Voice Assistant Using Natural Language Processing (NLP)	In this study, they have developed a voice assistant that can perform any kind of task in exchange for commands given by the users without any error. They have added more features like listening to the user's voice only and not being activated by environmental noise.	As it is written in this paper, our voice assistant does not have the feature of listening to the user's voice. We are currently working on this and will include it in the next updates of our software.
[3]	Desktop Assistant AI Using Python	They discussed a Python-based voice-activated personal assistant in this paper. This assistant currently works online and performs basic tasks like weather updates, and streaming music, searching Wikipedia, opening desktop applications, etc. The functionality of the current system is limited to working online only.	In comparison to their voice assistant, our voice assistant does not work online. Currently, we are working on that feature. And we will roll it in the coming updates of our voice assistant.
[4]	JARVIS: A PC Voice Assistant	This voice assistant has automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, and translating words from one language to another language, accessing youtube videos, sending mail through voice, and solving computational queries.	JARVIS and our project is kind of the same but does not have the features like retrieving weather forecast details and translating words from one language to another language. This is a minor difference and we would like to include these features in the coming updates.



[9]	Personal Assistant with Voice Recognition Intelligence	This paper focuses on "PARI", which is specially designed to help Native and Blind people who work on their Voice Commands. It also has the capability of recognizing voice commands without an internet connection. It has various functionalities for mobile devices, like network connection and managing various applications with just voice commands. It contains key features like Voice Pattern Detection, Keyword Learning, etc.	PARI is a brilliant voice assistant that is specially designed for blind people. It is a great initiative and is very helpful software. Our voice assistant does not focus on this but we would like to add these extra features in our project so that it can be beneficial to the blind as well as disabled people.
[12]	Vitro: Designing a Voice Assistant for the Scientific Lab Workplace	In this paper, they have designed a voice assistant and also done research on how the voice assistant can play a major role in a laboratory.	Here they have built a smart voice assistant that is specifically used for Scientific Lab Workspace. In comparison, our voice assistant is not designed to work in the laboratory but we have this concept on our mind. This is a new concept, so maybe this concept can further be included in our project.
[13]	Firefox Voice: An Open and Extensible Voice Assistant Built Upon the Web	This paper mainly focuses on "Firefox Voice", a voice assistant which is developed by the Mozilla Foundation and its subsidiary, the Mozilla Corporation. It shows how the voice assistant works and what its features are.	Now the voice assistant over here is mainly designed for a web browser. In comparison, our voice assistant is not designed for a web browser and also does not work online. We are currently working on this and this feature will be seen in the coming updates of our software.
[15]	Home Automation using Arduino and Smart Phone	It is a home automation system using the Arduino Uno board and wireless fidelity technology. It accepts commands only through clicks. Home automation through Android mobile is designed for physically challenged and disabled people.	This is a home automation and it is specially designed for physically challenged and disabled people using Arduino. In comparison, our voice assistant is not that advanced and is also not built for physically challenged and disabled people.
[16]	A voice based text mail system for the visually impaired	Here they have proposed an android application designed specifically for visually challenged people. It provides a voice-based mailing service where they can read and send mail on their own, without any guidance. The users have to use certain keywords which will perform certain actions, e.g., read, send, compose mail, address book, etc. This email system can be used by a blind person to access mail easily and efficiently.	Here we have a text-based mail system that is specifically built for visually impaired people. In comparison to their voice assistant, our system does not provide voice-based mail delivery and is also not built for visually impaired people. We are currently working on getting these features onboard as soon as possible. It will be rolled in further updates of our voice assistant.
[17]	Voice Control Human Assistance Robot	The robot developed in this project is able to move in any direction, like the front, back, left, or right, according to the voice commands received from the user through a microphone as part of our hardware in this project. There is	The project that we have made is different than theirs because we have not developed a robot system. They have developed an advanced robot that is controlled by voice commands. We have this project on our mind

		an autonomous voice command which can instantly make the robot move automatically without hitting any obstacle using an ultrasonic sensor. This device will help users give a uniform look to their lawn with ease. As well, it can also be used for the blind or physically challenged people by embedding this system into the wheelchair, which will make an autonomous wheelchair.	and maybe in the future, our voice assistant can be incorporated into a robot system.
[19]	Voice Control Device using Raspberry Pi	This paper describes the working of a device based on the implementation of a voice command system as an intelligent personal assistant. This voice-driven device uses the Raspberry Pi as its main hardware. A speech-to-text engine is used to convert the voice command to simple text. Query processing is then applied using natural language processing (NLP) to this text to interpret the intended meaning of the command given by the user. After interpreting the intended meaning, text-to-speech conversion is used to give the appropriate output in the form of speech. The services provided by the device depending on the input given, such as weather, telling time, or accessing online applications to listen to music.	The device created here is an advanced voice-controlled device that is developed using raspberry pi. This device is way better than ours and can perform many different tasks than our voice assistant. It is an advanced technology and we can research this and make a working model in the upcoming updates of our software.
[21]	Voice Recognition based Intelligent Wheelchair and GPS Tracking System	Here is a development of a voice recognition-based intelligent wheelchair system for physically handicapped people who are unable to drive the wheelchair by hand. So this system works like this: the patient can operate the wheelchair using voice commands and the location of the patient can be tracked using a GPS module in the wheelchair that tracks and sends the information to a smartphone application (app) via Firebase. The Voice Module V3 is used to record a patient's voice and recognize that voice to follow the instructions of the patient.	This is an intelligent wheelchair and GPS tracking system which can be used by physically handicapped people. This is a great initiative by this team and we are looking for this technology for physically handicapped people. We can take points from this project and implement these in the coming updates of our system.

IX. CONCLUSION

The virtual assistant we have created is able to do almost everything that the user commands it to do from opening a particular file on the system to web surfing to gather or collect information on the required topic. We kept a simple approach to our problem using python. Some main Python packages used in our product are this are speech Recognition, Python PyAudio, and Python TTS. We have successfully made a working virtual assistant which can be activated by the user using the wake keyword "SARA", and can manipulate the system using verbal commands. It eases most of the tasks of the user like searching the web, accessing youtube videos, sending mail through voice, etc.



In the future, we hope to incorporate more Artificial Intelligence into our project, such as Machine Learning, Neural networks, and so on, as well as the Internet of Things. With the addition of these elements, we will be able to improve our voice assistant by adding new features to it.

REFERENCES

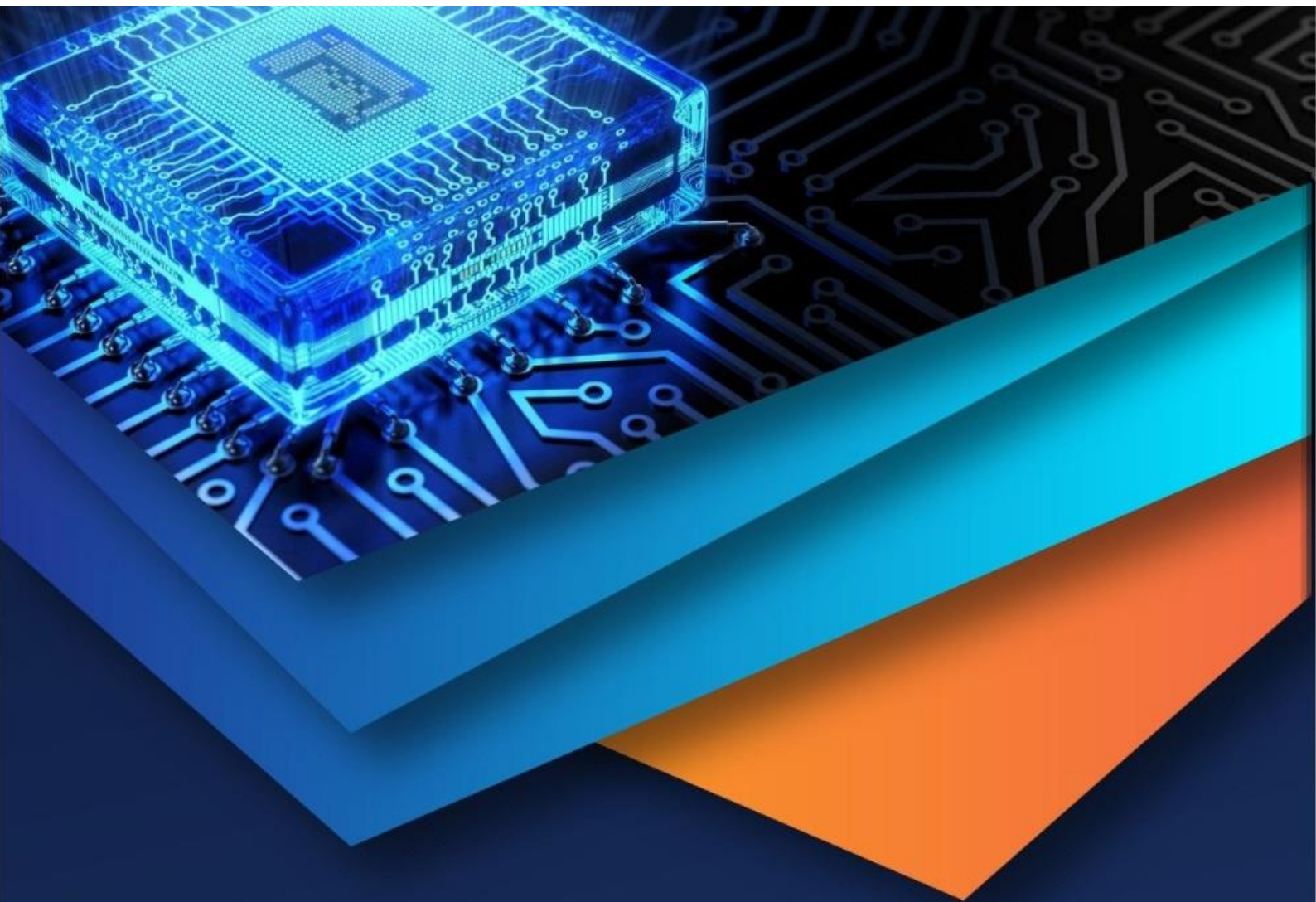
- [1] Jaydeep, Dr. P. A. Shewale, E. Bhushan, A. Fernandes, and R. Khartadkar. "A Voice Based Assistant Using Google Dialogflow and MachineLearning." International Journal of Scientific Research in Science and Technology 8, no. 3 (2021): 06-17.
- [2] Kumar, Lalit. "Desktop Voice Assistant Using Natural Language Processing (NLP)." International Journal for Modern Trends in Science and Technology (2020): n. pag.
- [3] International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Volume 6, Issue 2, June 2021 "Desktop Assistant AI Using Python"
- [4] Vora, Jash, Deepak Yadav, Ronak Jain, and Jaya Gupta. "JARVIS: A PC Voice Assistant." (2021).
- [5] Geetha, V., C. K. Gomathy, Kottamasu Manasa Sri Vardhan, and Nukala Pavan Kumar. "The Voice Enabled Personal Assistant for Pc using Python."
- [6] Pandey, Ankit, Vaibhav Vashist, Prateek Tiwari, Sunil Sikka, and Priyanka Makkar. "Smart Voice Based Virtual Personal Assistants with Artificial Intelligence." Artificial & Computational Intelligence/Published Online: June (2020).
- [7] July 2021| IJIRT | Volume 8 Issue 2 | ISSN: 2349-6002 "Voice Assistant Using Python." Nivedita Singh, Dr. Diwakar Yagyasen, Mr. Surya Vikram Singh, Gaurav Kumar, Harshit Agrawal
- [8] Shende, Deepak, Ria Umahiya, Monika Raghorte, Aishwarya Bhisikar, and Anup Bhange. "AI Based Voice Assistant Using Python." Journal of Emerging Technologies and Innovative Research 6, no. 2 (2019): 506-509.
- [9] Kulhalli, Kshama V., Kotrappa Sirbi, and Mr Abhijit J. Patankar. "Personal assistant with voice recognition intelligence." International Journal of Engineering Research and Technology 10, no. 1 (2017).
- [10] Patil, Akshay, Suyash Samant, Mohit Ramtekkar, Shubham Ragaji, and Jayashree Khanapuri. "Intelligent Voice Assistant." In Proceedings of the 3rd International Conference on Advances in Science & Technology (ICAST). 2020.
- [11] International Journal of Research in Engineering and Science (IJRES) ISSN (Online): 2320-9364, ISSN (Print): 2320-9356 www.ijres.org Volume 10 Issue 2 I 2022 I PP. 15-20 "Research Paper onDesktop Voice Assistant."
- [12] Cambre, Julia, Ying Liu, Rebecca E. Taylor, and Chinmay Kulkarni. "Vitro: Designing a Voice Assistant for the Scientific Lab Workplace." In Proceedings of the 2019 on Designing Interactive Systems Conference, pp. 1531-1542. 2019.
- [13] Cambre, Julia, Alex C. Williams, Afsaneh Razi, Ian Bicking, Abraham Wallin, Janice Tsai, Chinmay Kulkarni, and Jofish Kaye. "Firefox Voice: An Open and Extensible Voice Assistant Built Upon the Web." In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, pp. 1-18. 2021.
- [14] Popović, Branislav, Edvin Pakoci, Nikša Jakovljević, Goran Kočič, and Darko Pekar. "Voice assistant application for the Serbian language." In 2015 23rd Telecommunications Forum Telfor (TELFOR), pp. 858-861. IEEE, 2015.
- [15] Mr. T. M.Senthil Ganesan, M. Rama Jothi, R. S. Sangavi, L. Umayal, 2019 "Home Automation using Arduino and Smart Phone" INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ETEDM
- [16] M.J, Carmel Mary Belinda, N. Rupavathy and Mahalakshmi N.R. "A voice based text mail system for visually impaired." International journal of engineering and technology 7 (2018): 132.
- [17] John, Linda, Nilesh Vishwakarma, and Rajat Sharma. "Voice Control Human Assistance Robot." In National Conference on Technical Advancements for Social Upliftment, Proceedings of the 2 nd VNC. 2020.
- [18] Dias, Pubudu M., and Kithsiri Jayakody. "Virtual Assistant in Native Language." In 2020 IEEE Asia-Pacific Conference on Geoscience, Electronics and Remote Sensing Technology (AGERS), pp. 16-18. IEEE, 2020.
- [19] Singh, Pooja, Pinki Nayak, Arpita Datta, Depanshu Sani, Garima Raghav, and Rahul Tejpal. "Voice Control Device using Raspberry Pi." In 2019 Amity International Conference on Artificial Intelligence (AICAI), pp. 723-728. IEEE, 2019.
- [20] Joy, Jerry, Aparna Kannan, Shreya Ram, and S. Rama. "Speech Emotion Recognition using Neural Network and MLPClassifier." IJES, April-2020 (2020).
- [21] Aktar, Nasrin, Israt Jaharr, and Bijoya Lala. "Voice recognition based intelligent wheelchair and GPS tracking system." In 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), pp. 1-6. IEEE, 2019.

OTHER REFERENCES

- [1] Cambre, Julia, Alex C. Williams, Afsaneh Razi, Ian Bicking, Abraham Wallin, Janice Tsai, Chinmay Kulkarni, and Jofish Kaye. "Firefox Voice: An Open and Extensible Voice Assistant Built Upon the Web." In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, pp. 1-18. 2021.
- [2] Liao, Song, Christin Wilson, Long Cheng, Hongxin Hu, and Huixing Deng. "Measuring the effectiveness of privacy policies for voice assistant applications." In Annual Computer Security Applications Conference, pp. 856-869. 2020.
- [3] Pérez, Anxo, Paula Lopez-Otero, and Javier Parapar. "Designing an Open Source Virtual Assistant." Multidisciplinary Digital Publishing Institute Proceedings. Vol. 54. No. 1. 2020.
- [4] Braun, Michael, Anja Mainz, Ronee Chadowitz, Bastian Pflöging, and Florian Alt. "At your service: Designing voice assistant personalities to improve automotive user interfaces." In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, pp. 1-11. 2019.
- [5] Zhang, Ying, Mohammad Pezeshki, Philémon Brakel, Saizheng Zhang, Cesar Laurent Yoshua Bengio, and Aaron Courville. "Towards end-to-end speech recognition with deep convolutional neural networks." arXiv preprint arXiv:1701.02720 (2017).
- [6] Shalini, Shradha, Trevor Levins, Erin L. Robinson, Kari Lane, Geunhye Park, and Marjorie Skubic. "Development and comparison of customised voice-assistant systems for independent living older adults." In International Conference on Human-Computer Interaction, pp. 464-479. Springer, Cham, 2019.
- [7] Palanica, Adam, Anirudh Thommandram, Andrew Lee, Michael Li, and Yan Fossat. "Do you understand the words that are coming outta my mouth? Voice assistant comprehension of medication names." NPJ digital medicine 2, no. 1 (2019): 1-6.



- [8] Friedman, Natalie, Andrea Cuadra, Ruchi Patel, Shiri Azenkot, Joel Stein, and Wendy Ju. "Voice assistant strategies and opportunities for people with tetraplegia." In The 21st International ACM SIGACCESS Conference on Computers and Accessibility, pp. 575-577. 2019.
- [9] Marr B. Artificial intelligence in practice: how 50 successful companies used AI and machine learning to solve problems. John Wiley & Sons; 2019 May 28.
- [10] Braun, Michael, Anja Mainz, Ronee Chadowitz, Bastian Pfleging, and Florian Alt. "At your service: Designing voice assistant personalities to improve automotive user interfaces." In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, pp. 1-11. 2019.
- [11] Beirl, Diana, Y. Rogers, and Nicola Yuill. "Using voice assistant skills in family life." In Computer-Supported Collaborative Learning Conference, CSCL, vol. 1, pp. 96-103. International Society of the Learning Sciences, Inc., 2019.
- [12] Lötsch J, Ultsch A. Machine learning in pain research. Pain. 2018 Apr;159(4):623.
- [13] Winkler R, Söllner M. Unleashing the potential of chatbots in education: A state-of-the-art analysis. In Academy of Management Annual Meeting (AOM) 2018.
- [14] Kepuska V, Bohouta G. 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) 2018 Jan 8 (pp. 99-103). IEEE.
- [15] True, A. Miracle Made. "IBM's Watson Analytics for Health Care." (2017).
- [16] Thakur N, Hiwrale A, Selote S, Shinde A, Mahakalkar N. Artificially Intelligent Chatbot. Universal Research Reports. 2017;4(6):43.
- [17] Hill J, Ford WR, Farreras IG. Real conversations with artificial intelligence: A comparison between human-human online conversations and human-chatbot conversations. Computers in human behaviour. 2015 Aug 1;49:245-50.
- [18] Noda K, Arie H, Suga Y, Ogata T. Multimodal integration learning of robot behaviour using deep neural networks. Robotics and Autonomous Systems. 2014 Jun 1;62(6):721-36.
- [19] Lei, Xin, Andrew Senior, Alexander Gruenstein, and Jeffrey Sorensen. "Accurate and compact large vocabulary speech recognition on mobile devices." (2013)
- [20] Aron, Jacob. "How innovative is Apple's new voice assistant, Siri?." (2011): 24.
- [21] Nguyen P, Heigold G, Zweig G. Speech recognition with flat direct models. IEEE Journal of Selected Topics in Signal Processing. 2010 Sep 27;4(6):994-1006.
- [22] Huang J, Zhou M, Yang D. Extracting Chatbot Knowledge from Online Discussion Forums. In IJCAI 2007 Jan 6 (Vol. 7, pp. 423-428).
- [23] Fryer L, Carpenter R. Bots as language learning tools. Language learning and technology. Language Learning & Technology. 2006;10(3):8-14.
- [24] Allen JB. From Lord Rayleigh to Shannon: How do humans decode speech?. In International Conference on Acoustics, Speech and Signal Processing 2002.
- [25] Atal B, Rabiner L. A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing. 1976 Jun;24(3):201-12.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)

Appendix B: Sample Code

```
import tkinter as tk
from tkinter import messagebox
from tkinter import ttk
import json
import pyttsx3
import speech_recognition as sr
from datetime import datetime
import sys
import openai

engine = pyttsx3.init()

openai.api_key =
"sk-853KaHuD783gMWO7UsIWt3BibkFJIE2zSV4o8vcTkFbMsCzh"

def speak_text(text):
    engine.say(text)
    engine.runAndWait()

def record_audio():
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        audio = recognizer.listen(source)

    try:
        transcription = recognizer.recognize_google(audio)
        return transcription.lower()
    except sr.UnknownValueError:
        print("Sorry, I didn't understand.")
    except sr.RequestError as e:
        print(f"Error occurred during transcription: {e}")

    return None
```

```

def add_task(task, tasks):
    current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    for i, (stored_task, timestamp) in enumerate(tasks):
        if task == stored_task:
            tasks[i] = (task, current_time) # Update the existing task with a new
timestamp
            break
        else:
            tasks.append((task, current_time))
    save_tasks(tasks) # Save tasks to file
    print(f"Task '{task}' added at {current_time}")
    speak_text(f"Task '{task}' added at {current_time}")

def check_task(task, tasks):
    for stored_task, timestamp in tasks:
        if task == stored_task:
            print(f"Task '{task}' was done at {timestamp}")
            speak_text(f"Task '{task}' was done at {timestamp}")
            return

    print(f"Task '{task}' was not done")
    speak_text(f"Task '{task}' was not done")

def save_tasks(tasks):
    with open("tasks.txt", "w") as file:
        for task, timestamp in tasks:
            file.write(f"{task},{timestamp}\n")

def load_tasks():
    tasks = []
    try:
        with open("tasks.txt", "r") as file:
            for line in file:
                task, timestamp = line.strip().split(",")
                tasks.append((task, timestamp))
    
```

```
except FileNotFoundError:
    print("No tasks found.")
return tasks
```

```
def reminder():
    tasks = load_tasks() # Load tasks from file
```

```
while True:
    # speak_text("How can I assist you?")
    user_input = record_audio()

    if user_input:
        if "add task" in user_input:
            speak_text("What task did you complete?")
            task_input = record_audio()
```

```
            if task_input:
                add_task(task_input, tasks)
```

```
        elif "check task" in user_input:
            speak_text("Which task would you like to check?")
            task_input = record_audio()
```

```
            if task_input:
                check_task(task_input, tasks)
```

```
        elif "goodbye" in user_input:
            speak_text("Goodbye!")
            break
```

```
def save_text_as_audio(response, audio_f):
    engine.save_to_file(response, audio_f)
    engine.runAndWait()
```

```
def transcribe_audio_to_text(filename):
    recognizer = sr.Recognizer()
    with sr.AudioFile(filename) as source:
        audio = recognizer.record(source)
    try:
        return recognizer.recognize_google(audio)
    except sr.UnknownValueError:
        print("Speech recognition could not understand audio")
    except sr.RequestError:
        print("Could not request results from speech recognition service")
    return ""
```

```
def generate_response(prompt):
    response = openai.Completion.create(
        engine="text-davinci-003",
        prompt=prompt,
        max_tokens=4000,
        stop=None,
    )
    return response.choices[0].text
```

```
def write_output_to_file(text, filename):
    with open(filename, "w") as file:
        file.write(text + "\n")
```

```
def VA():
    while True:
        print("Say 'Julia' to ask questions")
        recognizer = sr.Recognizer()
```

```

with sr.Microphone() as source:
    audio = recognizer.listen(source)
try:
    transcription = recognizer.recognize_google(audio)
    if transcription.lower() == "julia":
        filename = "C:\\Users\\govind kiran\\Desktop\\project\\output.txt"
        audio_f = "C:\\Users\\govind kiran\\Desktop\\project\\audio.mp3"
        print("Ask a question")
        with sr.Microphone() as source:
            print("Listening...")
            recognizer.adjust_for_ambient_noise(source)
            audio = recognizer.listen(source, phrase_time_limit=None,
timeout=None)

        with open(filename, "wb") as f:
            f.write(audio.get_wav_data())

        text = transcribe_audio_to_text(filename)
        if text:
            print(f"You said: {text}")
            if text.lower().startswith("write"):
                response = generate_response(text)
                print(f"Julia says: {response}")
                speak_text(response)
                write_output_to_file(response, filename)
            elif text.lower().startswith("record"):
                response = generate_response(text)
                print(f"Julia says: {response}")
                speak_text(response)
                save_text_as_audio(response, audio_f)
            elif text.lower().startswith("rap"):
                text="I want you to act as a rapper. You will come up with
powerful and meaningful lyrics. Your lyrics should have an intriguing
meaning and message which people can relate too.It should be short. My
first request is “ I need a rap song about"+text+"."

```



```

        response = generate_response(text)
        print(f"Julia says: {response}")
        speak_text(response)
        write_output_to_file(response, filename)
        save_text_as_audio(response, audio_f)
    elif text.lower().startswith("tell"):
        text="I want you to act as a storyteller for kids. You will come
up with entertaining stories that are engaging, imaginative and captivating
for kids. It can be fairy tales, educational stories or any other type of stories
which has the potential to capture their attention and imagination.Since this
is for kids, it should be short. My first request is '"+text+".'"
        response = generate_response(text)
        print(f"Julia says: {response}")
        speak_text(response)
        write_output_to_file(response, filename)
        save_text_as_audio(response, audio_f)
    elif text.lower().startswith("pronounce"):
        text= "I want you to act as an English pronunciation
assistant . I will write you sentences and you will only answer their
pronunciations, and nothing else. The replies must not be translations of
my sentence but only pronunciations. Do not write explanations on replies.
My first sentence is '"+text+"'"
        response = generate_response(text)
        print(f"Julia says: {response}")
    elif text.lower().startswith("meaning"):
        text="I want you to act as a dictionary.Word is '"+text+".'"
        response = generate_response(text)
        print(f"Julia says: {response}")
        speak_text(response)
    elif text.lower().startswith("goodbye"):
        speak_text("Goodbye!")
        break
    else :
        response = generate_response(text)
        print(f"Julia says: {response}")

```

```
        speak_text(response)
    except sr.UnknownValueError:
        print("Speech recognition could not understand audio")
    except sr.RequestError:
        print("Could not request results from speech recognition service")
    except Exception as e:
        print(f"An error occurred: {e}")
```

```
class Redirect():
```

```
    def __init__(self, widget):
        self.widget = widget

    def write(self, text):
        self.widget.insert('end', text)
        self.widget.see('end')
```

```
def load_user_data():
    try:
        with open('user_data.json', 'r', encoding='utf-8') as file:
            user_data = json.load(file)
    except FileNotFoundError:
        user_data = {}
    return user_data
```

```
def save_user_data(user_data):
    with open('user_data.json', 'w', encoding='utf-8') as file:
        json.dump(user_data, file)
```

```
def validate_login():
    username = entry_username.get()
    password = entry_password.get()

    user_data = load_user_data()
```

```
    if username in user_data and user_data[username] == password:
        messagebox.showinfo("Login Successful", "Welcome, admin!" if
username == "admin" else f"Welcome, {username}!")
        if username == "admin":
            show_main_page(admin=True)
        else:
            show_main_page()
    else:
        messagebox.showerror("Login Failed", "Invalid username or
password.")
```

```
def show_main_page(admin=False):
    window.withdraw()
    main_window.deiconify()
```

```
    if admin:
        button_check_users.pack(side=tk.LEFT, padx=10)
```

```
def logout():
    main_window.withdraw()
    window.deiconify()
```

```
def open_julia_page():
    main_window.withdraw()
    julia_window.deiconify()
```

```
def open_reminder_page():
    main_window.withdraw()
    reminder_window.deiconify()
```

```
def open_user_list():
    main_window.withdraw()
    user_list_window.deiconify()
```

```

def back_to_main():
    julia_window.withdraw()
    reminder_window.withdraw()
    user_list_window.withdraw()
    main_window.deiconify()

def validate_signup():
    global entry_signup_username, entry_signup_password
    username = entry_signup_username.get()
    password = entry_signup_password.get()

    if not username or not password:
        messagebox.showerror("Sign Up Failed", "Please enter a username
and password.")
        return

    user_data = load_user_data()

    if username in user_data:
        messagebox.showerror("Sign Up Failed", "Username already exists.
Please choose a different username.")
    else:
        user_data[username] = password
        save_user_data(user_data)
        messagebox.showinfo("Sign Up Successful", "Account created
successfully. You can now log in.")

def show_signup_page():
    global signup_window, entry_signup_username, entry_signup_password
    window.withdraw()
    signup_window = tk.Toplevel(window)
    signup_window.title("Sign Up")

```

```
signup_window.geometry(f'{window_width}x{window_height}+{x_coordinate}+{y_coordinate}')
```

```
signup_frame = tk.Frame(signup_window)
signup_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
label_signup_username = tk.Label(signup_frame, text="Username:")
label_signup_username.pack()
```

```
entry_signup_username = tk.Entry(signup_frame)
entry_signup_username.pack()
```

```
label_signup_password = tk.Label(signup_frame, text="Password:")
label_signup_password.pack()
```

```
entry_signup_password = tk.Entry(signup_frame, show="*")
entry_signup_password.pack()
```

```
button_signup_submit = tk.Button(signup_frame, text="Sign Up",
command=validate_signup)
button_signup_submit.pack(pady=10)
```

```
button_signup_back = tk.Button(signup_frame, text="Back",
command=back_to_login)
button_signup_back.pack(pady=10)
```

```
def back_to_login():
    signup_window.withdraw()
    window.deiconify()
```

```
# Create the main window
window = tk.Tk()
window.title("Login Page")
```

```
# Set window dimensions for an average phone screen
window_width = 320
window_height = 480
screen_width = window.winfo_screenwidth()
screen_height = window.winfo_screenheight()
x_coordinate = int((screen_width / 2) - (window_width / 2))
y_coordinate = int((screen_height / 2) - (window_height / 2))
window.geometry(f'{window_width}x{window_height}+{x_coordinate}+{y_co
ordinate}')
```

```
# Create the main page window
main_window = tk.Toplevel(window)
main_window.title("Main Page")
main_window.geometry(f'{window_width}x{window_height}+{x_coordinate}
+{y_coordinate}')
main_window.withdraw()
```

```
# Create the Julia window
julia_window = tk.Toplevel(window)
julia_window.title("Julia Page")
julia_window.geometry(f'{window_width}x{window_height}+{x_coordinate}+
{y_coordinate}')
julia_window.withdraw()
```

```
# Create the Reminder window
reminder_window = tk.Toplevel(window)
reminder_window.title("Reminder Page")
reminder_window.geometry(f'{window_width}x{window_height}+{x_coordin
ate}+{y_coordinate}')
reminder_window.withdraw()
```

```
# Create the User List window
user_list_window = tk.Toplevel(window)
user_list_window.title("User List")
```

```
user_list_window.geometry(f'{window_width}x{window_height}+{x_coordinate}+{y_coordinate}')
user_list_window.withdraw()
```

```
# Create and position the widgets for the login page
login_frame = tk.Frame(window)
login_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
label_username = tk.Label(login_frame, text="Username:")
label_username.pack()
```

```
entry_username = tk.Entry(login_frame)
entry_username.pack()
```

```
label_password = tk.Label(login_frame, text="Password:")
label_password.pack()
```

```
entry_password = tk.Entry(login_frame, show="*")
entry_password.pack()
```

```
button_login = tk.Button(login_frame, text="Login",
                           command=validate_login)
button_login.pack(pady=10)
```

```
button_signup = tk.Button(login_frame, text="Sign Up",
                           command=show_signup_page)
button_signup.pack(pady=10)
```

```
# Create and position the widgets for the main page
main_frame = tk.Frame(main_window)
main_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
button_julia = tk.Button(main_frame, text="Julia",
                          command=open_julia_page)
button_julia.pack(pady=10)
```

```
button_reminder = tk.Button(main_frame, text="Reminder",  
command=open_reminder_page)  
button_reminder.pack(pady=10)
```

```
button_story = tk.Button(main_frame, text="StoryTeller",  
command=open_julia_page)  
button_story.pack(pady=10)
```

```
button_pronoun = tk.Button(main_frame, text="Pronouncer",  
command=open_julia_page)  
button_pronoun.pack(pady=10)
```

```
button_rap = tk.Button(main_frame, text="Rapper",  
command=open_julia_page)  
button_rap.pack(pady=10)
```

```
button_Dict = tk.Button(main_frame, text="Dictionary",  
command=open_julia_page)  
button_Dict.pack(pady=10)
```

```
button_logout = tk.Button(main_frame, text="Log Out", command=logout)  
button_logout.pack(pady=10)
```

```
button_check_users = tk.Button(main_frame, text="Check Users",  
command=open_user_list)  
button_check_users.pack(side=tk.LEFT, padx=10)
```

```
# Create and position the widgets for the Julia page  
julia_frame = tk.Frame(julia_window)  
julia_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
julia_output = tk.Text(julia_frame, height=10, width=30)  
julia_output.pack(pady=10)
```



```
button_start_julia = tk.Button(julia_frame, text="Start", command=VA)
button_start_julia.pack(pady=10)
```

```
button_back_julia = tk.Button(julia_frame, text="Back",
command=back_to_main)
button_back_julia.pack(pady=10)
```

```
# Create and position the widgets for the Reminder page
reminder_frame = tk.Frame(reminder_window)
reminder_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
reminder_output = tk.Text(reminder_frame, height=10, width=40)
reminder_output.pack(pady=10)
```

```
button_start_reminder = tk.Button(reminder_frame, text="Start",
command=reminder)
button_start_reminder.pack(pady=10)
```

```
button_back_reminder = tk.Button(reminder_frame, text="Back",
command=back_to_main)
button_back_reminder.pack(pady=10)
```

```
# Create and position the widgets for the User List page
user_list_frame = tk.Frame(user_list_window)
user_list_frame.place(relx=0.5, rely=0.5, anchor="center")
```

```
user_list_table = ttk.Treeview(user_list_frame, columns=('Username',
'Password'))
user_list_table.heading('#0', text='ID')
user_list_table.heading('Username', text='Username')
user_list_table.heading('Password', text='Password')
user_list_table.pack(pady=10)
```

```
def load_users():  
    user_data = load_user_data()  
    user_list_table.delete(*user_list_table.get_children())  
    for i, (username, password) in enumerate(user_data.items(), start=1):  
        user_list_table.insert("", 'end', text=str(i), values=(username,  
password))
```

```
load_users()
```

```
button_back_users = tk.Button(user_list_frame, text="Back",  
command=back_to_main)  
button_back_users.pack(pady=10)
```

```
# Start the main loop  
window.mainloop()
```

Appendix C: CO-PO And CO-PSO Mapping

COURSE OUTCOMES:

After completion of the course the student will be able to

SL. NO	DESCRIPTION	Blooms' Taxonomy Level
CO1	Identify technically and economically feasible problems (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO2	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO3	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions of minimal complexity by using modern tools & advanced programming techniques (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO4	Prepare technical report and deliver presentation (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO5	Apply engineering and management principles to achieve the goal of the project (Cognitive Knowledge Level: Apply)	Level 3: Apply

CO-PO AND CO-PSO MAPPING

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	3	3	3	3		2	2	3	2	2	2	3	2	2	2
CO2	3	3	3	3	3	2		3	2	3	2	3	2	2	2
CO3	3	3	3	3	3	2	2	3	2	2	2	3			2
CO4	2	3	2	2	2			3	3	3	2	3	2	2	2
CO5	3	3	3	2	2	2	2	3	2		2	3	2	2	2

3/2/1: high/medium/low

JUSTIFICATIONS FOR CO-PO MAPPING

MAPPING	LOW/ MEDIUM/ HIGH	JUSTIFICATION
100003/CS6 22T.1-PO1	HIGH	Identify technically and economically feasible problems by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS6 22T.1-PO2	HIGH	Identify technically and economically feasible problems by analysing complex engineering problems reaching substantiated conclusions using first principles of mathematics.
100003/CS6 22T.1-PO3	HIGH	Design solutions for complex engineering problems by identifying technically and economically feasible problems.
100003/CS6 22T.1-PO4	HIGH	Identify technically and economically feasible problems by analysis and interpretation of data.
100003/CS6 22T.1-PO6	MEDIUM	Responsibilities relevant to the professional engineering practice by identifying the problem.
100003/CS6 22T.1-PO7	MEDIUM	Identify technically and economically feasible problems by understanding the impact of the professional engineering solutions.
100003/CS6 22T.1-PO8	HIGH	Apply ethical principles and commit to professional ethics to identify technically and economically feasible problems.
100003/CS6 22T.1-PO9	MEDIUM	Identify technically and economically feasible problems by working as a team.
100003/CS6 22T.1-PO10	MEDIUM	Communicate effectively with the engineering community by identifying technically and economically feasible problems.
100003/CS6 22T.1-PO11	MEDIUM	Demonstrate knowledge and understanding of engineering and management principles by selecting the technically and economically feasible problems.
100003/CS6 22T.1-PO12	HIGH	Identify technically and economically feasible problems for long term learning.
100003/CS6 22T.1-PSO1	MEDIUM	Ability to identify, analyze and design solutions to identify technically and economically feasible problems.
100003/CS6 22T.1-PSO2	MEDIUM	By designing algorithms and applying standard practices in software project development and Identifying technically and economically feasible problems.
100003/CS6 22T.1-PSO3	MEDIUM	Fundamentals of computer science in competitive research can be applied to Identify technically and economically feasible problems.
100003/CS6 22T.2-PO1	HIGH	Identify and survey the relevant by applying the knowledge of mathematics, science, engineering fundamentals.

100003/CS6 22T.2-PO2	HIGH	Identify, formulate, review research literature, and analyze complex engineering problems get familiarized with software development processes.
100003/CS6 22T.2-PO3	HIGH	Design solutions for complex engineering problems and design based on the relevant literature.
100003/CS6 22T.2-PO4	HIGH	Use research-based knowledge including design of experiments based on relevant literature.
100003/CS6 22T.2-PO5	HIGH	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes by using modern tools.
100003/CS6 22T.2-PO6	MEDIUM	Create, select, and apply appropriate techniques, resources, by identifying and surveying the relevant literature.
100003/CS6 22T.2-PO8	HIGH	Apply ethical principles and commit to professional ethics based on the relevant literature.
100003/CS6 22T.2-PO9	MEDIUM	Identify and survey the relevant literature as a team.
100003/CS6 22T.2-PO10	HIGH	Identify and survey the relevant literature for a good communication to the engineering fraternity.
100003/CS6 22T.2-PO11	MEDIUM	Identify and survey the relevant literature to demonstrate knowledge and understanding of engineering and management principles.
100003/CS6 22T.2-PO12	HIGH	Identify and survey the relevant literature for independent and lifelong learning.
100003/CS6 22T.2-PSO1	MEDIUM	Design solutions for complex engineering problems by Identifying and survey the relevant literature.
100003/CS6 22T.2-PSO2	MEDIUM	Identify and survey the relevant literature for acquiring programming efficiency by designing algorithms and applying standard practices.
100003/CS6 22T.2-PSO3	MEDIUM	Identify and survey the relevant literature to apply the fundamentals of computer science in competitive research.
100003/CS6 22T.3-PO1	HIGH	Perform requirement analysis, identify design methodologies by using modern tools & advanced programming techniques and by applying the knowledge of mathematics, science, engineering fundamentals.
100003/CS6 22T.3-PO2	HIGH	Identify, formulate, review research literature for requirement analysis, identify design methodologies and develop adaptable & reusable solutions.

100003/CS6 22T.3-PO3	HIGH	Design solutions for complex engineering problems and perform requirement analysis, identify design methodologies.
100003/CS6 22T.3-PO4	HIGH	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS6 22T.3-PO5	HIGH	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools.
100003/CS6 22T.3-PO6	MEDIUM	Perform requirement analysis, identify design methodologies and assess societal, health, safety, legal, and cultural issues.
100003/CS6 22T.3-PO7	MEDIUM	Understand the impact of the professional engineering solutions in societal and environmental contexts and Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS6 22T.3-PO8	HIGH	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions by applying ethical principles and commit to professional ethics.
100003/CS6 22T.3-PO9	MEDIUM	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS6 22T.3-PO10	MEDIUM	Communicate effectively with the engineering community and with society at large to perform requirement analysis, identify design methodologies.
100003/CS6 22T.3-PO11	MEDIUM	Demonstrate knowledge and understanding of engineering requirement analysis by identifying design methodologies.
100003/CS6 22T.3-PO12	HIGH	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS6 22T.3-PSO3	MEDIUM	The ability to apply the fundamentals of computer science in competitive research and prior to that perform requirement analysis, identify design methodologies.
100003/CS6 22T.4-PO1	MEDIUM	Prepare technical report and deliver presentation by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS6 22T.4-PO2	HIGH	Identify, formulate, review research literature, and analyze complex engineering problems by preparing technical report and deliver presentation.

100003/CS6 22T.4-PO3	MEDIUM	Prepare Design solutions for complex engineering problems and create technical report and deliver presentation.
100003/CS6 22T.4-PO4	MEDIUM	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions and prepare technical report and deliver presentation.
100003/CS6 22T.4-PO5	MEDIUM	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and Prepare technical report and deliver presentation.
100003/CS6 22T.4-PO8	HIGH	Prepare technical report and deliver presentation by applying ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
100003/CS6 22T.4-PO9	HIGH	Prepare technical report and deliver presentation effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS6 22T.4-PO10	HIGH	Communicate effectively with the engineering community and with society at large by prepare technical report and deliver presentation.
100003/CS6 22T.4-PO11	MEDIUM	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work by prepare technical report and deliver presentation.
100003/CS6 22T.4-PO12	HIGH	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by prepare technical report and deliver presentation.
100003/CS6 22T.4-PSO1	MEDIUM	Prepare a technical report and deliver presentation to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas.
100003/CS6 22T.4-PSO2	MEDIUM	To acquire programming efficiency by designing algorithms and applying standard practices in software project development and to prepare technical report and deliver presentation.
100003/CS6 22T.4-PSO3	MEDIUM	To apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs by preparing technical report and deliver presentation.
100003/CS6 22T.5-PO1	HIGH	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS6 22T.5-PO2	HIGH	Identify, formulate, review research literature, and analyze complex engineering problems by applying engineering and management principles to achieve the goal of the project.

100003/CS6 22T.5-PO3	HIGH	Apply engineering and management principles to achieve the goal of the project and to design solutions for complex engineering problems and design system components or processes that meet the specified needs.
100003/CS6 22T.5-PO4	MEDIUM	Apply engineering and management principles to achieve the goal of the project and use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS6 22T.5-PO5	MEDIUM	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and to apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO6	MEDIUM	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities by applying engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO7	MEDIUM	Understand the impact of the professional engineering solutions in societal and environmental contexts, and apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO8	HIGH	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice and to use the engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO9	MEDIUM	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings and to apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO11	MEDIUM	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments and to apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PO12	HIGH	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change and to apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PSO1	MEDIUM	The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas. Apply engineering and management principles to achieve the goal of the project.

100003/CS6 22T.5-PSO2	MEDIUM	The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry and to apply engineering and management principles to achieve the goal of the project.
100003/CS6 22T.5-PSO3	MEDIUM	The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur and apply engineering and management principles to achieve the goal of the project.

