**VOICE ASSISTANT**

**A MINI PROJECT REPORT**

**SUBMITTED BY**

**G.AISHWARYA(953622104003)**

**P.PAVITRA SREE(953622104074)**

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**RAMCO INSTITUTE OF TECHNOLOGY,**

**RAJAPALAYAM**

**JANUARY 2024**

**CS3361 DATA SCIENCE LABORATORY**

**BONAFIDE CERTIFICATE**

Certified that this mini project report “**VOICE ASSISTANT**” is the bonafide work of “**AISHWARYA G**(953622104003), **P.PAVITRA SREE** (953622104074)”, who carried out the mini project work under my supervision.

**SIGNATURE**  **SIGNATURE**

S.Vijaya Amala Devi M.E Dr.K.Vijayalakshmi M.E., Ph.D

**FACULTY IN-CHARGE HEAD OF THE DEPARTMENT**

Department of computer science and Department of computer science and

Engineering, Engineering,

Ramco Institute of technology, Ramco Institute of technology,

North Venganallur Village, North Venganallur Village,

Rajapalayam-626117. Rajapalayam-626117.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

Voice assistants have transformed how we engage with technology, offering spoken language interpretation and responses in both text and voice formats. This abstract explores the benefits of multimodal voice assistants, which provide users with the option to receive responses in either text or synthesized speech.

Multimodal voice assistants offer flexibility, catering to individual preferences and situational contexts. Users can choose between reading text responses for clarity or listening to voice output for a more engaging interaction. This versatility enhances accessibility, particularly for those with visual impairments or in noisy environments.

In addition to enhancing user experience, multimodal voice assistants also facilitate more inclusive and accessible interactions. By offering options for both text and voice output, these systems accommodate a wider range of users with diverse communication preferences, disabilities, or language proficiencies. This inclusivity aligns with principles of universal design, ensuring that technology remains accessible and beneficial to all individuals, regardless of their abilities or backgrounds.

By integrating both text and voice output, multimodal voice assistants improve communication effectiveness and user engagement. Users can reinforce comprehension by accessing information through multiple sensory modalities, facilitating learning and memory retention.

Moreover, these assistants promote inclusivity by accommodating diverse communication preferences and abilities. By offering options for both text and voice output, they ensure accessibility for users of varying backgrounds and language proficiencies.

In summary, multimodal voice assistants represent a significant advancement in human-computer interaction, providing users with enhanced flexibility, clarity, and inclusivity in their interactions with technology.

**TABLE OF CONTENT:**

|  |  |  |
| --- | --- | --- |
| S.NO | CONTENT | PAGE NO |
| 1 | Introduction  1.1Voice Assistant  1.2Project Objective  1.3Project Specification | 5 |
| 2 | System Specification  2.1Hardware specification  2.2Software specification | 5 |
| 3 | Packages  3.1PYTTSX3  3.2WIKIPEDIA | 6 |
| 4 | Appendix  4.1Source code  4.2Explanation  4.3Screenshot | 7 |
| 5 | Conclusion | 9 |
| 6 | Futurework | 10 |
| 7 | Reference | 11 |

**1.INTRODUCTION**

In recent years, voice assistant technology has emerged as a transformative force in human-computer interaction, offering users a seamless and intuitive way to engage with digital devices and services. This abstract focuses on the integration of voice assistants within a movie recommendation system, exploring how this integration enhances user experience and accessibility. By providing users with the ability to interact with the system using natural language input and receive responses in both text and synthesized speech formats, this project aims to revolutionize the way users discover and engage with movies.

**1.1 Voice Assistant:**

Voice assistants are intelligent software applications that interpret and respond to spoken commands or queries from users. By leveraging natural language processing and machine learning algorithms, these assistants can understand user intent and provide relevant information or perform tasks accordingly. In the context of this project, the voice assistant serves as the primary interface through which users interact with the movie recommendation system, offering a convenient and hands-free way to access personalized movie suggestions.

**1.2 Project Objective:**

The primary objective of this project is to enhance the user experience of the movie recommendation system by integrating a voice assistant interface. By enabling users to interact with the system using voice commands, the project aims to streamline the process of discovering and selecting movies based on individual preferences and interests. Additionally, the project seeks to improve accessibility for users with disabilities or those who prefer voice-based interactions over traditional text-based interfaces.

**1.3 Project Specification:**

The project involves integrating a voice assistant interface into the existing movie recommendation system, enabling users to access personalized movie suggestions through natural language input. The voice assistant will be capable of understanding user queries related to movie preferences, genres, actors, and other relevant criteria. It will then process these queries, retrieve relevant movie recommendations from the system's database, and present them to the user in both text and synthesized speech formats. Furthermore, the project will ensure compatibility with various devices and platforms to maximize accessibility and user reach.

**2.SYSTEM SPECIFICATION**

**2.1 Hardware specification**

* Processor : Intel dual core
* Processor speed: 1.04GHZ
* Ram : 1GB
* Monitor
* Keyboard

**2.2** **Software** **specification**

* Language : Python
* Compiler : Idle

**3.PACKAGES**

**3.1 PYTTSX3**

* Pyttsx3 helps Python talk.
* It turns written words into spoken words.
* Works on different computers.
* Can change how it sounds.
* Easy to use for making computers talk.

**INSTALLING PYTTSX3 PACKAGE**

Pip install pyttsx3

**Why use pyttsx3**

Pyttsx3 is a Python tool that turns written words into spoken words. It's handy for making apps easier to use for people who can't read well or who prefer listening. It's also great for making voice-controlled assistants or adding speech to games and stories. Overall, Pyttsx3 makes it simple to give your computer or device a voice.

**3.2 WIKIPEDIA**

* Wikipedia is a free book online.
* People write and change its pages.
* It talks about many things.
* You can use it for free.
* It's in many languages.
* Some people check if it's true.

**INSTALLING WIKIPEDIA PACKAGE:**

Pip install Wikipedia

**Why use Wikipedia:**

Wikipedia is like a big online book that anyone can read and write. It talks about lots of different things, like history, animals, and famous people. It's free for everyone to use, which is really cool. People from all over the world write and change its pages, so you can learn from many different perspectives. You can read Wikipedia in lots of languages too, which makes it even more useful. Some people check to make sure the information is true, so it's usually a reliable place to learn new things. Overall, Wikipedia is a handy tool for getting information on just about anything you're curious about.

**4.APPENDIX**

**4.1 SOURCE CODE**

import pyttsx3

import wikipedia

voice=pyttsx3.init()

In=input("searching:")

result=wikipedia.summary(In,sentences=3)

print(result)

voice.say(result)

voice.runAndWait()

**4.2 EXPLANATION**

**import pyttsx3:** This line imports the pyttsx3 module, which is a Python library used for text-to-speech conversion.

**import wikipedia:** This line imports the wikipedia module, which provides access to Wikipedia articles and information.

**voice = pyttsx3.init():** This line initializes a new instance of the pyttsx3 engine and assigns it to the variable voice. This engine will be responsible for converting text to speech.

**In = input("searching:"):** This line prompts the user to input a search query by displaying the message "searching:". The input text entered by the user is then stored in the variable In.

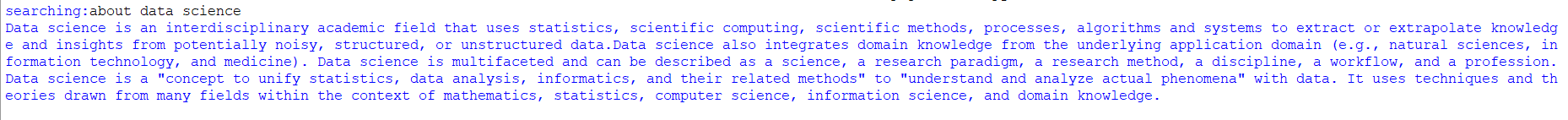
**result = wikipedia.summary(In, sentences=3):** This line retrieves a summary of the Wikipedia article related to the search query provided by the user (In). The summary() function from the wikipedia module is used, and the sentences=3 argument specifies that only the first 3 sentences of the summary should be returned. The summary is stored in the variable result.

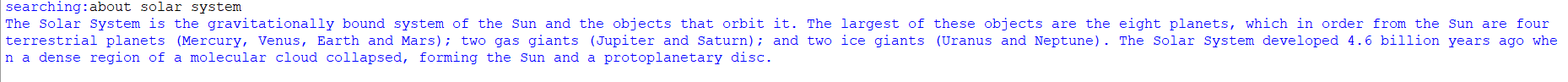
**print(result):** This line prints the summary of the Wikipedia article retrieved in the previous step to the console.

**voice.say(result):** This line instructs the voice engine to speak out the contents of the result variable, which contains the summary of the Wikipedia article.

**voice.runAndWait():** This line ensures that the voice engine waits until the speech is finished before continuing with the execution of the program. This allows the spoken text to be fully heard before the program exits.

**4.3 SCREENSHOT**

****

****

It also gives voice as output along with the text.

**5. CONCLUSION**

In addition to its core functionality, the code showcases several notable features and possibilities. Firstly, it highlights the seamless integration of Python libraries, namely pyttsx3 and wikipedia, demonstrating the versatility and power of Python for various tasks. Moreover, it exemplifies the potential for natural language processing and human-computer interaction, as users can input search queries in plain language, receiving concise summaries of Wikipedia articles in response. Furthermore, the code underscores the accessibility benefits of text-to-speech technology, making information available audibly to individuals with visual impairments or those who prefer auditory learning. Additionally, it hints at the potential for further expansion and customization, such as incorporating error handling to gracefully manage cases where Wikipedia articles are not found or refining the user interface for a more polished user experience. Overall, the code serves as a foundation for building more sophisticated applications leveraging text-to-speech conversion and web scraping capabilities in Python.

**6.FUTURE WORK**

In future iterations, several enhancements and refinements could be implemented to further improve the functionality and usability of the code. One potential avenue for future work is the incorporation of error handling mechanisms to gracefully manage scenarios where the Wikipedia article corresponding to the user's input is not found or where there are network connectivity issues.

Additionally, the code could be extended to allow users to specify the number of sentences they want in the summary or to provide options for different language preferences. Furthermore, integrating natural language processing (NLP) techniques could enhance the search functionality by enabling more sophisticated query understanding and article summarization. Another potential enhancement could involve the development of a graphical user interface (GUI) to provide a more intuitive and user-friendly interaction experience.

Additionally, exploring ways to optimize and streamline the text-to-speech conversion process could improve the responsiveness and efficiency of the application. Finally, considering the integration of other sources of information beyond Wikipedia could broaden the scope and utility of the application, providing users with access to a wider range of knowledge resources.

**7.REFERENCE**

<https://www.geeksforgeeks.org/python-text-to-speech-by-using-pyttsx3/>

<https://www.geeksforgeeks.org/wikipedia-module-in-python/>

|  |  |
| --- | --- |
| **PERFORMANCE** |  |
| **VIVA VOCE** |  |
| **MINIPROJECT** |  |
| **TOTAL** |  |