DOCUMENTATION

There are 8 files(modules) present in the main package of this project. These files are named as follows: -

1. VoiceAssistant\_main.py
2. speakListen.py
3. websiteWork.py
4. textRead.py
5. dictator.py
6. menu.py
7. speechtotext.py
8. TextToSpeech.py

A combination of all these modules makes the Voice Assistant work efficiently.

VoiceAssistant\_main.py

This is the main file that encapsulates the other 7 files. It is advisable to run this file to avail all the benefits of the Voice Assistant.

After giving the command to run this file to your computer, you will have to say “**Hello Python**” to activate the voice assistant. After the activation, a menu will be displayed on the screen, showing all the tasks that Voice Assistant can do. This menu is displayed with the help of the print\_menu() function present in the menu.py module.

Speaking out the respective commands of the desired task will indicate the Voice Assistant to do the following task. Once the speech is recorded, it will be converted to ` str ` by hear() or short\_hear() function of the speakListen.py module.

For termination of the program after a task is complete, the command “**close python**” should be spoken. For abrupt termination of the program, for Windows and Linux – The ctrl + c key combination should be used.

speakListen.py

This is the module that contains the following functions: -

1. speak(text) – This function speaks out the ‘text’ provided as a parameter. The text is a string(str). They say() and runAndWait() functions of Engine class in pyttsx3 enable the assistant to speak. Microsoft ***SAPI5*** has provided the voice.
2. hear() – This function records the voice for 9 seconds using your microphone as source and converts speech to text using recognize\_google(). recognize\_google() performs speech recognition on ``audio\_data`` (an ``AudioData`` instance), using the Google Speech Recognition API.
3. long\_hear(duration\_time) – This function records voice for the ‘duration\_time’ provided with 60 seconds as the default time. It too converts speech to text in a similar fashion to hear()
4. short\_hear(duration\_time) – This functions records voice similar to hear() but for 5 seconds.
5. greet(g) - Uses the datetime library to generate current time and then greets accordingly.

websiteWork.py

This module mainly handles this project's ‘searching on the web’ task. It uses wikipedia and webbrowser libraries to aid its tasks. Following are the functions present in this module: -

1. google\_search() – Searches the sentence spoken by the user on the web and opens the google-searched results in the default browser.
2. wiki\_search() - Searches the sentence spoken by the user on the web and opens the Wikipedia-searched results in the default browser. It also speaks out the summary of the result and asks the user whether he wants to open the website of the corresponding query.

textRead.py

This module is mainly related to file processing and converting text to speech. Following are the functions present in this module: -

1. ms\_word – Read out the TEXT in MS Word (.docx) file provided in the location.
2. pdf\_read – Can be used to read pdf files and more specifically eBooks. It has 3 options
   1. Read a single page
   2. Read a range of pages
   3. Read a lesson
   4. Read the whole book

It can also print the index and can find out the author’s name, the title, and the total number of pages in the PDF file.

1. doubleslash(location) – Mainly intended to help Windows users, if the user copies the default path containing 1 ‘/ ’; the program doubles it so it is not considered an escape sequence.
2. print\_index(toc) - Prints out the index in proper format with the title name and page number. It takes ‘toc’ as a parameter which is a nested list with toc[i][1] - Topic name and toc[i][2] – with page number.
3. print\_n\_speak\_index(toc) - It is similar to print\_index(), but it also speaks out the index here.
4. book\_details(author, title, total\_pages) - Creates a table of book details like author name, title, and total pages. It uses table and console from rich library.

**IMPORTANT: The voice assistant asks you the location of your file to be read by it. It won’t detect the file if it is present in the OneDrive folder or any other protected or third-party folder. Also, it would give an error if the extension of the file is not provided.**

**For example; Consider a docx file ‘**abc**’ and pdf file ‘**fgh**’ present in valid directories and folders named ‘**folder\_loc’**.**

**When location is fed as ‘**folder\_loc \abc**’ or ‘**folder\_loc\fgh’ **it gives an error,**

**but if the location is given as** ‘folder\_loc \abc.docx’ **or ‘**folder\_loc \fgh.pdf’**, then it won’t give an error.**

dictator.py

This module is like the dictator function and dictates the text that we speak. So basically, it converts the speech that we speak to text. The big\_text(duration\_time) function encapsulates the long\_hear() function. So by default, it records for 60 seconds but it can record for any amount of time as specified by the user.

menu.py

It prints out the menu which contains the tasks and their corresponding commands. The rich library is being used here.