REPORT

ON

VOICE VIRTUAL ASSISTANT

TO BE DEVELOPED TO FULFILL THE REQUIREMENTS FOR

MAJOR PROJECT (BCA)

Department of Computer Applications
Chitkara University, Punjab

under the supervision of

1)Dr Jaiteg Singh
Professor

2)Dr Rajesh Kaushal
Associate Professor

Submitted by :-

1)Shubham Sharma
Roll No.2010992598

BCA
(Batch 2020-23)

CERTIFICATE OF APPROVAL.

The undersigned certify that they have read and recommended to the department of Computer Application for acceptance, a project report entitled “Voice Virtual Assistant” submitted by 2010992598(Shubham Sharma),

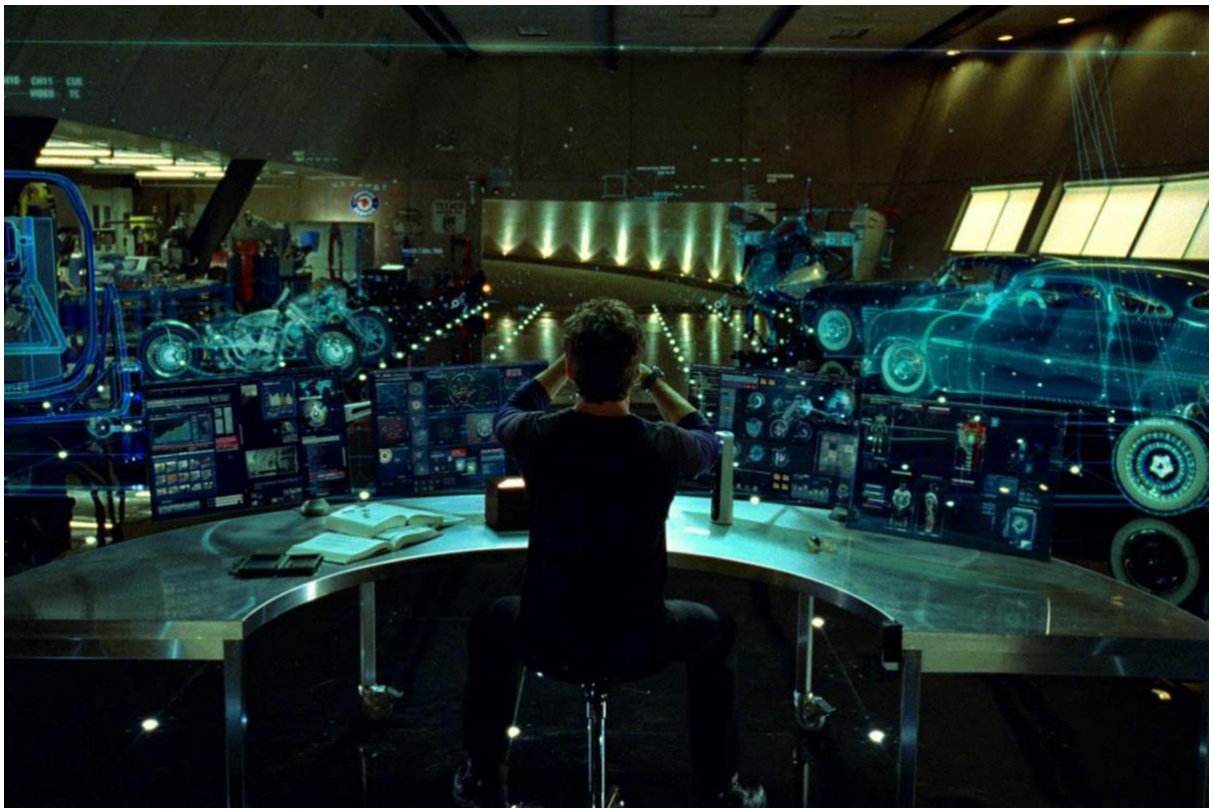
(Mr Jaswinder Singh)
Associate Professor Department

DECLARATION

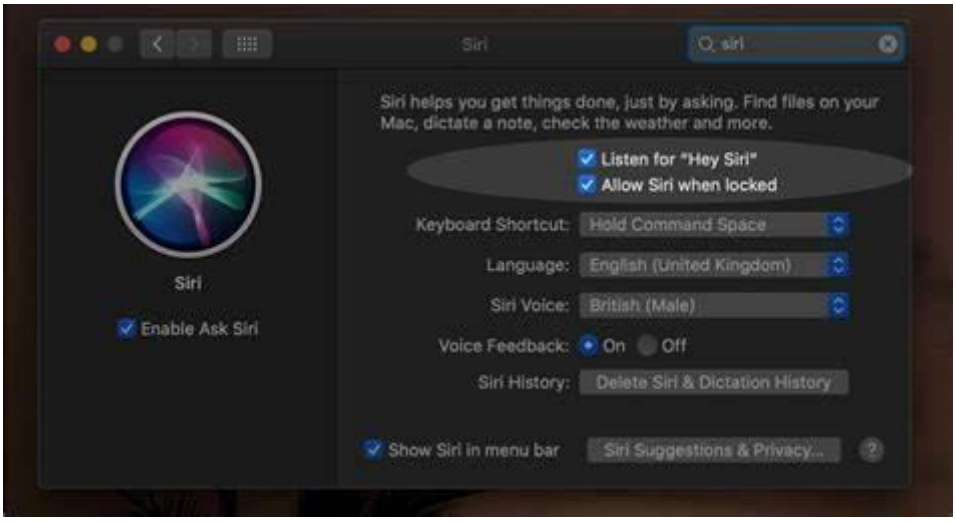
I Shubham Sharma a student of "Bachelor of Computer Application", (2020-23), Chitkara University, Punjab hereby inform that the work presented in this dissertation entitled " Voice Virtual Assistant" is outcome of my own work , is correct to the best of my knowledge and this work has been carried out taking care engineering ethics. The work presented does not infringe any patent work and has not been submitted to any other university or anywhere else for award of any degree or any professional diploma.

1)Abstract:-

Virtual based ai assistant that can answer and perform every sort of task from reading and sending mails to setting up an alarm, notifying your day to day task.



CASE STUDY-SIRI (APPLE AI ASSISTANT)



Siri is a voice-activated personal assistant that was first introduced by Apple in 2011. The application was created to be a smart assistant to help users with a variety of tasks, such as making phone calls, sending messages, setting reminders, and finding information. Siri uses natural language processing and machine learning algorithms to understand and respond to user requests. One of the key challenges for Siri was to create an intuitive and easy-to-use interface for users to interact with. Apple focused on making the interface as simple as possible, allowing users to ask Siri questions or give commands in a conversational manner. Siri uses artificial intelligence and machine learning algorithms to understand and interpret natural language, making it easier for users to get the information they need. Another challenge for Siri was to integrate with other Apple applications seamlessly. Siri can interact with many of Apple's native apps, including Calendar, Reminders, and Maps, as well as third-party applications, making it a versatile personal assistant for users.One of the biggest successes of Siri is its integration with HomeKit, Apple's smart home platform. Siri can be used to control smart home devices, such as lights, thermostats, and door locks, with simple voice commands. This integration has made Siri an important part of the smart home ecosystem. Siri has also been continuously updated and improved since its launch, with new features being added regularly. Some notable updates include the ability to translate between languages, send and receive money through Apple Pay, and answer more complex questions.

One of the biggest criticisms of Siri has been its limited functionality and accuracy compared to other virtual assistants, such as Amazon's Alexa and Google Assistant. However, Apple has been working to address these issues with recent updates, including the introduction of Siri Shortcuts, which allows users to create custom commands and automate tasks.In conclusion, Siri has been a successful and innovative personal assistant application, which has had a significant impact on the virtual assistant market. While there have been some criticisms, Apple's continuous improvements and updates to Siri have made it a valuable tool for many users.

System Requiements:-

i)Product Definition:-

Our project that is based an ai based virtual based voice assistant , that takes voice command as input from the user then converts it to text, then compares it with the conditions that are given in the source, then give the output in the form of audio or text depending upon the condition.

a)Problem Statement:-

The problem statement of a virtual voice assistant is to provide a convenient and efficient way for users to interact with their devices and perform various tasks using voice commands. The main objective of a virtual voice assistant is to understand natural language and respond appropriately, enabling users to control their devices without the need for physical input. Some common tasks that virtual voice assistants can perform include setting reminders, sending messages, making phone calls, playing music, searching the internet, and controlling smart home devices. The challenge lies in developing an intelligent and accurate system that can interpret and process voice commands in a reliable and timely manner, while also ensuring privacy and security for the user.

b)Functions to be provided:-

A virtual voice assistant can provide a wide range of functions depending on its design and capabilities. Here are some of the common functions that virtual voice assistants can perform:

1. Set alarms and reminders
2. Make phone calls and send text messages
3. Play music and videos
4. Provide weather forecasts and news updates
5. Answer general knowledge questions

6. Control smart home devices, such as lights and thermostats
7. Provide navigation and directions
8. Schedule appointments and meetings
9. Set timers and countdowns
10. Make restaurant reservations and order food
11. Provide sports scores and updates
12. Read audiobooks and e-books
13. Translate languages
14. Provide health and fitness advice
15. Play games and quizzes
16. Search the internet for information
17. Dictate and send emails
18. Create shopping lists and make purchases
19. Tell jokes and provide entertainment
20. Assist with accessibility needs for people with disabilities.

These are just a few examples of the many functions that virtual voice assistants can provide. As technology continues to improve, the capabilities of virtual voice assistants are expected to expand even further.

c)Solution Strategy:-

The solution strategy for developing a virtual voice assistant typically involves the following steps:

1. Define the purpose and scope: Determine the specific functions and capabilities that the virtual voice assistant will perform. Identify the target audience and the devices the assistant will be compatible with.

2. Choose the appropriate platform and technology: Select the technology platform and tools that are most suitable for the development of the virtual voice assistant. Some popular platforms include Amazon Alexa, Google Assistant, and Apple Siri.

3. Develop the natural language processing (NLP) engine: NLP is the core technology that enables the virtual voice assistant to understand and interpret spoken language. The NLP engine should be able to accurately recognize speech, extract meaning from the words and phrases, and respond appropriately.

4. Build the dialogue management system: The dialogue management system controls the flow of conversation between the user and the virtual voice assistant. It should be designed to provide a natural and intuitive user experience, and be able to handle complex and multi-turn conversations.

5. Implement integration with external services: The virtual voice assistant should be able to connect with external services and APIs to provide information and perform tasks such as making reservations, ordering products, or controlling smart home devices.

6. Test and refine the system: Testing is an important part of the development process to identify and fix any issues or bugs. User feedback should be incorporated to improve the accuracy and effectiveness of the virtual voice assistant.

7. Continuously improve and update: The virtual voice assistant should be continuously updated to incorporate new features and functionality, and to keep up with changing user needs and technology trends. Regular maintenance and updates are essential to ensure that the assistant remains relevant and useful over time.

d)Acceptance Criteria:-

Acceptance criteria for a fully fledged build of a virtual voice assistant would depend on the specific requirements and goals of the project. However, here are some general criteria that could be considered:

1. Accuracy: The virtual voice assistant should be able to accurately recognize and interpret user input, including voice commands and natural language processing.

2. Functionality: The virtual voice assistant should be able to perform a wide range of functions, such as answering questions, setting reminders, making appointments, and controlling smart home devices.

3. Compatibility: The virtual voice assistant should be compatible with various operating systems, devices, and platforms.

4. Reliability: The virtual voice assistant should be reliable and consistent in its performance, with minimal downtime or errors.

5. Security: The virtual voice assistant should be designed with security in mind, with measures in place to protect user data and prevent unauthorized access.

6. Personalization: The virtual voice assistant should be able to learn and adapt to the preferences and behaviors of individual users, providing a personalized experience.

7. Integration: The virtual voice assistant should be able to integrate with other software and services, such as email, messaging apps, and third-party apps.

8. Accessibility: The virtual voice assistant should be accessible to users with disabilities, with features such as voice recognition, text-to-speech, and support for braille displays.

9. User Experience: The virtual voice assistant should provide a user-friendly and intuitive interface, with clear and concise responses and an overall pleasant experience for the user.

10. Scalability: The virtual voice assistant should be able to handle increasing volumes of users and data, with the ability to scale up or down as needed.

2). Feasibility Analysis:-

A feasibility analysis is an important step in determining the viability of any project, including a virtual voice assistant. In order to perform a feasibility analysis for a virtual voice assistant, here are some key factors to consider:

1. Technical Feasibility: This involves assessing whether the necessary technology is available to build the virtual voice assistant. This includes hardware and software requirements such as the appropriate programming languages, frameworks, and tools. It also includes considerations such as the availability of data and the ability to process natural language.
2. Market Feasibility: This involves analyzing the potential demand for the virtual voice assistant. You will need to identify your target audience and determine whether there is a sufficient market size to support the development and deployment of the assistant.
3. Financial Feasibility: This involves assessing whether the development and ongoing maintenance of the virtual voice assistant is financially viable. This includes the costs associated with hardware, software, and staffing, as well as revenue streams such as subscriptions or advertising.
4. Legal Feasibility: This involves assessing whether the virtual voice assistant complies with relevant laws and regulations. This may include data privacy laws, intellectual property laws, and consumer protection laws.
5. Operational Feasibility: This involves assessing whether the virtual voice assistant can be effectively integrated into the user's daily life. This includes considerations such as ease of use, reliability, and accessibility.

By conducting a comprehensive feasibility analysis, you can identify potential risks and opportunities associated with developing and deploying a virtual voice assistant. This will enable you to make informed decisions about the project and ensure its long-term success.

3)Project Plan:-

a)Team Structure:-

Our team has three members:-

1)Shubham Sharma(Team leader and responsible for execution):- Responsible for the development of the project ,including from taking suggestions from other members, to implementing them.

b)Development Process:-

We completed the project in three phases:-

First phase :-

Implementation of speak command , taking user command and giving output in the form of audio ,testing the working of pyttsx3 and sapi5, and and giving access of our microphone.

Second phase:-

Implementation of other functions such as the reading out live news , weather by implementing api keys from news and weather sites and with help of internet these keys will provide access to the latest news and weather condition, and many other functions such as searching on Wikipedia, telling advices etc.

Third phase :-

By developing and implementing a graphical user interface for our virtual voice assistant that interacts and starts and stops the microphone ,thus enabling and disabling access a speak command starts and terminates whole of the interface .

Programming language and development tools:-

1)Python:-

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

2)PyQt5:-

PyQt is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in. PyQt is free software developed by the British firm Riverbank Computing. It is available under similar terms to Qt versions older than 4.5; this means a variety of licenses including GNU General Public License (GPL) and commercial license, but not the GNU Lesser General Public License (LGPL).PyQt supports Microsoft Windows as well as various flavours of UNIX, including Linux and MacOS (or Darwin).

PyQt implements around 440 classes and over 6,000 functions and methods including:-

- A substantial set of GUI widgets
- Classes for accessing SQL databases (ODBC, MySQL, PostgreSQL, Oracle, SQLite)
- QScintilla, Scintilla-based rich text editor widget
- Data aware widgets that are automatically populated from a database
- An XML parser
- SVG support
- Classes for embedding ActiveX controls on Windows (only in commercial version)

PyQt5 contains the following Python modules:-

- QtCore:
  The QtCore module is the core module of PyQt5. It provides the fundamental functionality needed for all Qt applications, including the event loop, signals and slots mechanism, thread management, file I/O, and more. This module contains classes for handling time and date, managing resources, and interacting with the system's file system.

- QtGui:
  The QtGui module contains classes for creating and managing graphical user interfaces. It provides a set of essential widgets like buttons, labels, text fields, menus, and toolbars that form the foundation of any Qt application. Additionally, it offers advanced features such as drag and drop, graphics view, image manipulation, and input handling.

- QtWidgets:
  The QtWidgets module provides a set of high-level widgets and classes that build on top of the QtGui module. It offers more specialized widgets like dialogs, message boxes, progress bars, and tabbed interfaces, which are commonly used in desktop applications. This module also provides support for internationalization and accessibility, making it easy to create applications that can be used by people with disabilities or who speak different languages.

- QtMultimedia:
  The QtMultimedia module provides classes for multimedia programming, such as audio and video playback and recording. It includes classes for controlling cameras and microphones, as well as support for different multimedia formats like MP3, WAV, and AVI.

- QtNetwork:
  The QtNetwork module contains classes for network programming, including TCP and UDP sockets, HTTP and FTP protocols, and SSL/TLS encryption. It allows developers to write network-enabled applications with ease.

- QtOpenGL:
  The QtOpenGL module provides classes for OpenGL programming, allowing developers to create interactive 3D graphics. It supports advanced features such as shaders, textures, and framebuffers.

- QtPrintSupport:
  The QtPrintSupport module provides classes for printing, including printer and print dialog management. It allows developers to create print previews, set print options, and print documents in various formats.

- QtSql:
  The QtSql module provides classes for database programming, including support for SQL databases like MySQL and SQLite. It allows developers to interact with databases in a Qt-friendly way, including executing queries, retrieving results, and manipulating data.

- QtSvg:
  The QtSvg module provides classes for SVG (Scalable Vector Graphics) rendering and manipulation. It allows developers to load, display, and manipulate SVG files and graphics in their Qt applications.

- QtTest:
  The QtTest module provides classes for unit testing Qt applications. It includes classes for creating test cases, running tests, and reporting results.

- QtWebKit:
  The QtWebKit module provides a Qt-based web browser engine for rendering HTML, CSS, and JavaScript content. It allows developers to create web-enabled applications that can display web pages and interact with web-based services.

- QtXml:
  The QtXml module provides classes for XML parsing and serialization. It allows developers to read and write XML files, validate XML documents, and interact with XML-based web services.

- QtXmlPatterns:
  The QtXmlPatterns module provides classes for XSLT and XPath processing. It allows developers to transform XML data into different formats, filter and extract data, and validate XML documents against specific schemas.

- The uic module implements support for handling the XML files created by Qt Designer that describe the whole or part of a graphical user interface. It includes classes that load an XML file and render it directly, and classes that generate Python code from an XML file for later execution.

3)Sapi5:-

The Speech Application Programming Interface or SAPI is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications. To date, a number of versions of the API have been released, which have shipped either as part of a Speech SDK or as part of the Windows OS itself. Applications that use SAPI include Microsoft Office, Microsoft Agent and Microsoft Speech Server.

Test Plan:-

i) Functional, Performance, Stress tests :-

For a virtual voice assistant, the following functional, performance, and stress tests could be conducted:

Functional tests:-

1. Voice recognition accuracy test: This test verifies if the virtual voice assistant is able to accurately recognize the voice commands and respond appropriately.

2. Intent recognition test: This test verifies if the virtual voice assistant is able to understand the intent of the user's voice commands and respond appropriately.

3. Functionality test: This test verifies if the virtual voice assistant is able to perform the functions that it is designed for, such as setting reminders, making calls, sending messages, etc.

Performance tests:-

1. Response time test: This test measures the time taken by the virtual voice assistant to respond to user commands.

2. Concurrent user test: This test measures the performance of the virtual voice assistant when multiple users are using it simultaneously.

3. Network latency test: This test measures the performance of the virtual voice assistant in different network conditions, such as low bandwidth or high latency.

Stress tests:-

1. Load testing: This test measures the virtual voice assistant's ability to handle a large number of requests from multiple users simultaneously.

2. Capacity testing: This test measures the maximum capacity of the virtual voice assistant, in terms of the number of users it can handle at a time.

3. Endurance testing: This test measures the ability of the virtual voice assistant to handle continuous usage over an extended period of time, without any performance degradation or failure.

By conducting these tests, developers can ensure that the virtual voice assistant is functioning correctly, performing well under different conditions, and capable of handling extreme loads and stress.

Implementation / Conversion Plan:-

Implementing and converting to a virtual voice assistant involves a few key steps:

1. Define the requirements: First, you need to define the requirements for the virtual voice assistant. This includes the functions it should be able to perform, the languages it should support, and the platforms it should be available on.

2. Choose a platform: Next, you need to choose a platform for your virtual voice assistant. Popular platforms include Google Assistant, Amazon Alexa, Apple Siri, and Microsoft Cortana.

3. Develop the assistant: You will need to develop the virtual voice assistant, including designing the user interface, creating the voice recognition and natural language processing algorithms, and integrating it with other systems.

4. Test the assistant: Before launching the virtual voice assistant, it is important to test it thoroughly. This includes conducting functional, performance, and stress tests to ensure that it is working correctly and performing well under different conditions.

5. Launch the assistant: Once the virtual voice assistant has been developed and tested, it can be launched for use by customers.

6. Monitor and improve the assistant: After launching the virtual voice assistant, it is important to monitor its performance and user feedback. This will help you identify any issues or areas for improvement, and make necessary changes to the assistant.

7. Promote and market the assistant: Finally, you will need to promote and market the virtual voice assistant to attract users and increase adoption. This can include advertising, social media marketing, and partnerships with other companies or platforms.

Overall, implementing and converting to a virtual voice assistant requires careful planning, development, testing, and promotion to ensure a successful launch and adoption by users.

2)Introduction:-

2.1 Project Aim and Objectives:-

Aim:-To develop a virtual ai assistant using python that can take

Simple voice commands

Objectives:-

Perquisites:-

1)Make install a module called pyttsx3 to run general voice commands, define a function speak , it is a text-to-speech library.

2)Then after that install pypiwin32, it provides access to much of the Win32 API, the ability to create and use COM objects, and the Pythonwin environment.

3)Then after that we need to install sapi5 ,Microsoft Speech API (SAPI5) is the technology for voice recognition and synthesis provided by Microsoft. Starting with Windows XP, it ships as part of the Windows OS.

4) What Is VoiceId?

It is component that helps in selecting voices

- id helps us to select different voices.
- voice[0].id = Male voice
- voice[1].id = Female voice

Technology to be Used:-

Python, audiopy

Hardware and Software Requirement;-

Software Requirement:-

- **Windows 10 or 11 depending upon the editor requirement**
- **Vsc editor or pycharm depending upon the individual preference**
- **Pyttsx3 plugin to run voice commands**
- pypiwin32 based on 32 bit configuration
- Microsoft speech api (SAPI 5) is the technology for voice recognition and synthesis provided by Microsoft.

Hardware Requirement :-

- **RAM:-**

  8 GB ram
  DDR4 3200 MHz

- **SSD:-**

  512 GB nvme

- **Processor (CPU):-**

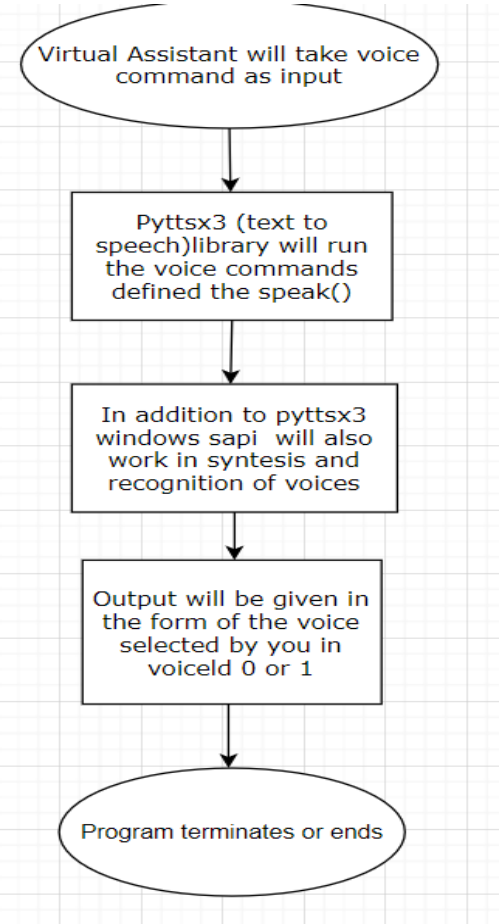
  AMD Ryzen 5
  4000 series
  4600H

- **GPU:-**

  Nvidia GTX 1650 4GB VRAM

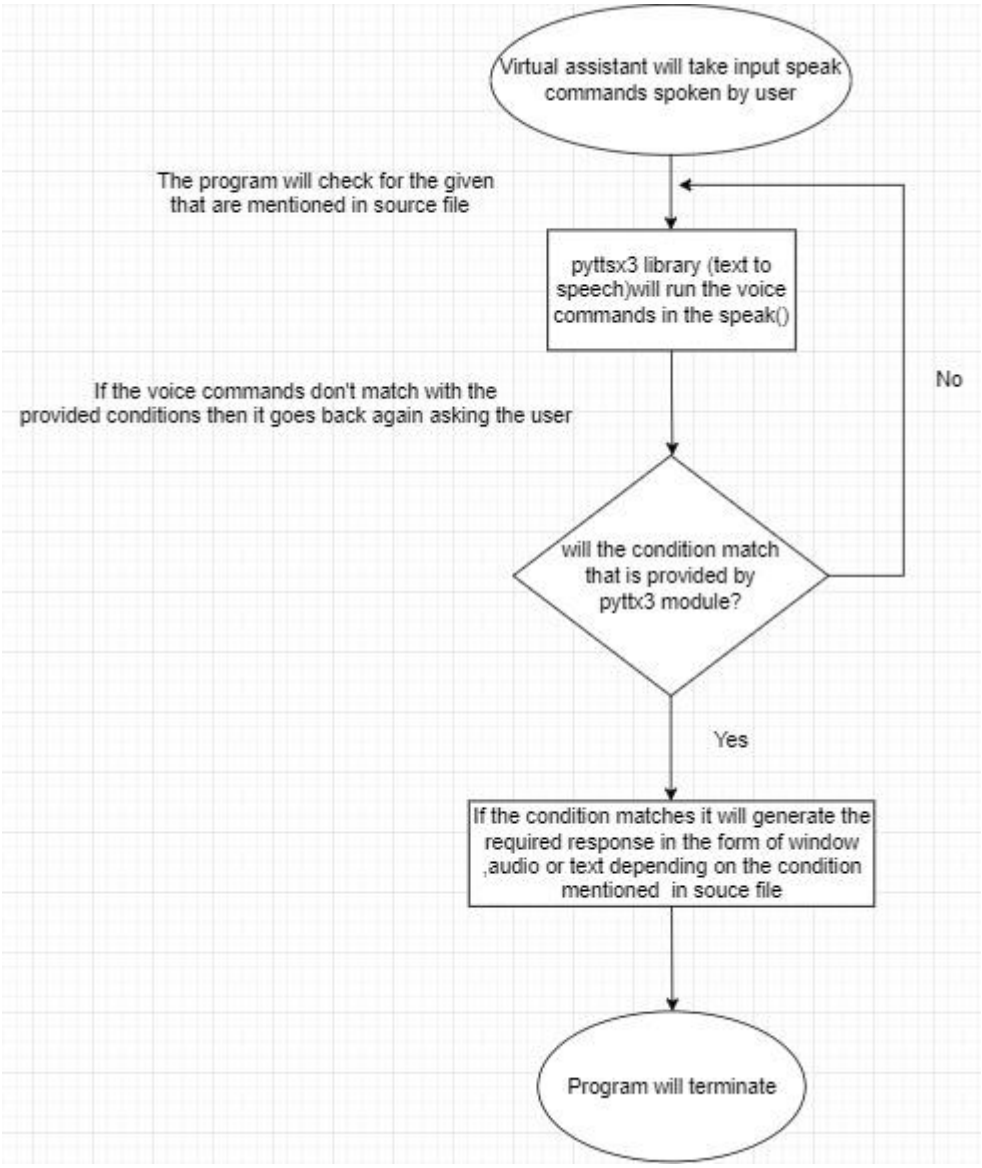
System Analysis ;-

Software Requirement Specification



This is a general flow chart depicting the working of virtual voice assistant.

DFD(Data Flow Diagram):-



Explanation:-

This DFD shows how the how the data is flowing in our virtual voice assistant , starting from taking the voice command from the user and then pyttsx3 module works , converts the data that is taken in the form of audio is converted to text. Then after that data is matched with the conditions, if the conditions match with the data then respective audio or text output depending on the condition otherwise the process is again started from the very beginning , till the condition is matched or the program can also terminate.

Existing compared to Proposed System:-

Existing technology such as google voice assistant , siri, alexa, cortana etc provide us with many functions such as day to day news ,weather temp etc but is time consuming and lack many features that we have covered in this project .

Functional and Performance Specifications Our project is fully functional performance oriented, the user can speak that will be taken as input function and will be matched as from the predefined function consisting from opening of browser such as chrome to opening command prompt cmd etc plus other features Such as online browsing of songs ,to giving access to our system inc/dec system volume.

Developing Environments:-

For developing and getting the most of the project we are performing various manual testing techniques. Also the project is developed using python,Pyaudio,pyttsx3,sapi5,pypiwin32 etc

Python:-

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Pyttsx3:-

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is

female and the second is male which is provided by “sapi5” for windows. It supports three TTS engines :

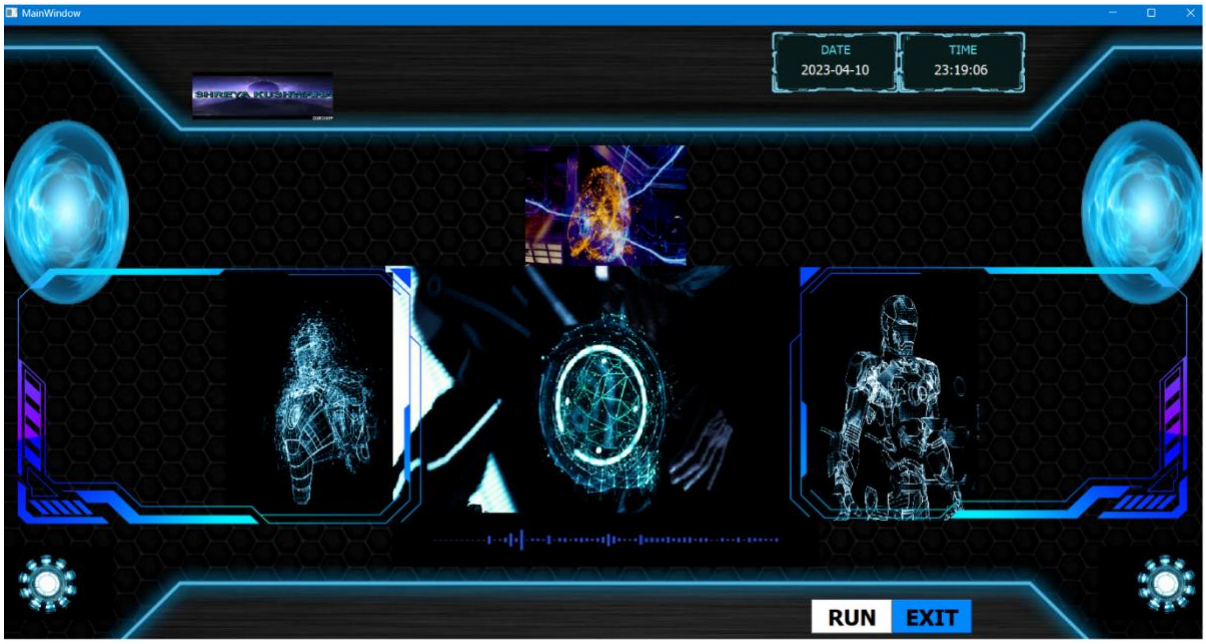
sapi5 – SAPI5 on Windows
nsss – NSSpeech Synthesizer on Mac OS X
espeak – eSpeak on every other platform

Sapi5:-

Microsoft Speech API (SAPI5) is the technology for voice recognition and synthesis provided by Microsoft. Starting with Windows XP, it ships as part of the Windows OS. If you are using a different OS, please consult the Microsoft Speech Technologies or the Speech SDK 5.1 sites (see addresses below).

Design :-

A svg file with background consisting of different ui component, audio input command welcoming , answering to question in response to the information mentioned in the program.



This interface is developed using qtdesigner that is a module of PyQt5:-

- Qt Designer is a visual interface design tool that is included with PyQt5. It allows developers to create GUI applications by designing and editing graphical user interfaces visually, without the need for manual coding. Qt Designer enables developers to create interfaces for their PyQt5 applications quickly and easily.

- Qt Designer provides a WYSIWYG (What You See Is What You Get) interface that allows developers to drag and drop GUI components onto a canvas to create the user interface. The components include buttons, labels, text fields, sliders, menus, and many others. Developers can also customize

the appearance and behavior of these components using properties editors, which are provided by the tool.

- One of the key benefits of Qt Designer is its ability to separate the visual design of the user interface from the code that implements its behavior. This is achieved through the use of an XML-based file format called UI files. Developers can create the user interface using Qt Designer and then use PyQt5 to load and interact with the UI file in their code. This approach allows developers to focus on the logic of the application, without being distracted by the details of the user interface.

- Qt Designer also supports a feature called signals and slots, which is a way of connecting events generated by the GUI components to specific functions or methods in the code. With signals and slots, developers can define the behavior of the application in response to user interactions with the GUI components, such as button clicks or menu selections.

- Another feature of Qt Designer is its integration with Qt Creator, an integrated development environment (IDE) for developing Qt applications. Qt Designer can be launched directly from within Qt Creator, making it easy to switch between designing the user interface and coding the application logic.

- In summary, Qt Designer is a powerful visual interface design tool that enables developers to create graphical user interfaces for their PyQt5 applications

quickly and easily. Its WYSIWYG interface, support for signals and slots, and integration with Qt Creator make it an essential tool for any PyQt5 developer.

We achieved this interface in the following steps:-

- To create an interface for a virtual voice assistant named Jarvis with only two buttons (start and stop) and various futuristic UI and SVG components using Qt Designer, you can follow the steps below:

- 1. Open Qt Designer and create a new form. To create a new form, go to File > New File or press Ctrl+N.

- In the New Form dialog box, select Dialog with Buttons Bottom from the list of form types, and then click Create.

- Add two QPushButton widgets to the form, one for the Start button and one for the Stop button.

- Use the SVG Editor (available in the Qt Designer) to add futuristic SVG components to the form. Some examples of SVG components that you can add include futuristic icons, animations, and graphics.

- Add a QLabel widget to the form to display the status of the virtual voice assistant, such as "Jarvis is currently active" or "Jarvis is currently inactive".

- 6. Use the Layout > Lay Out Horizontally and Layout > Lay Out Vertically options from the toolbar to align the widgets horizontally and vertically.

- Customize the appearance of the widgets using the properties editor. For example, you can change the font, size, and color of the text on the buttons and labels, as well as the background color of the form.

- Save the form as a .ui file.

- In your PyQt5 application, create a new Python file and import the necessary modules, including PyQt5.QtWidgets, PyQt5.uic, and PyQt5.QtCore.

- In the main function of your application, use the uic.loadUi method to load the .ui file and display the interface.

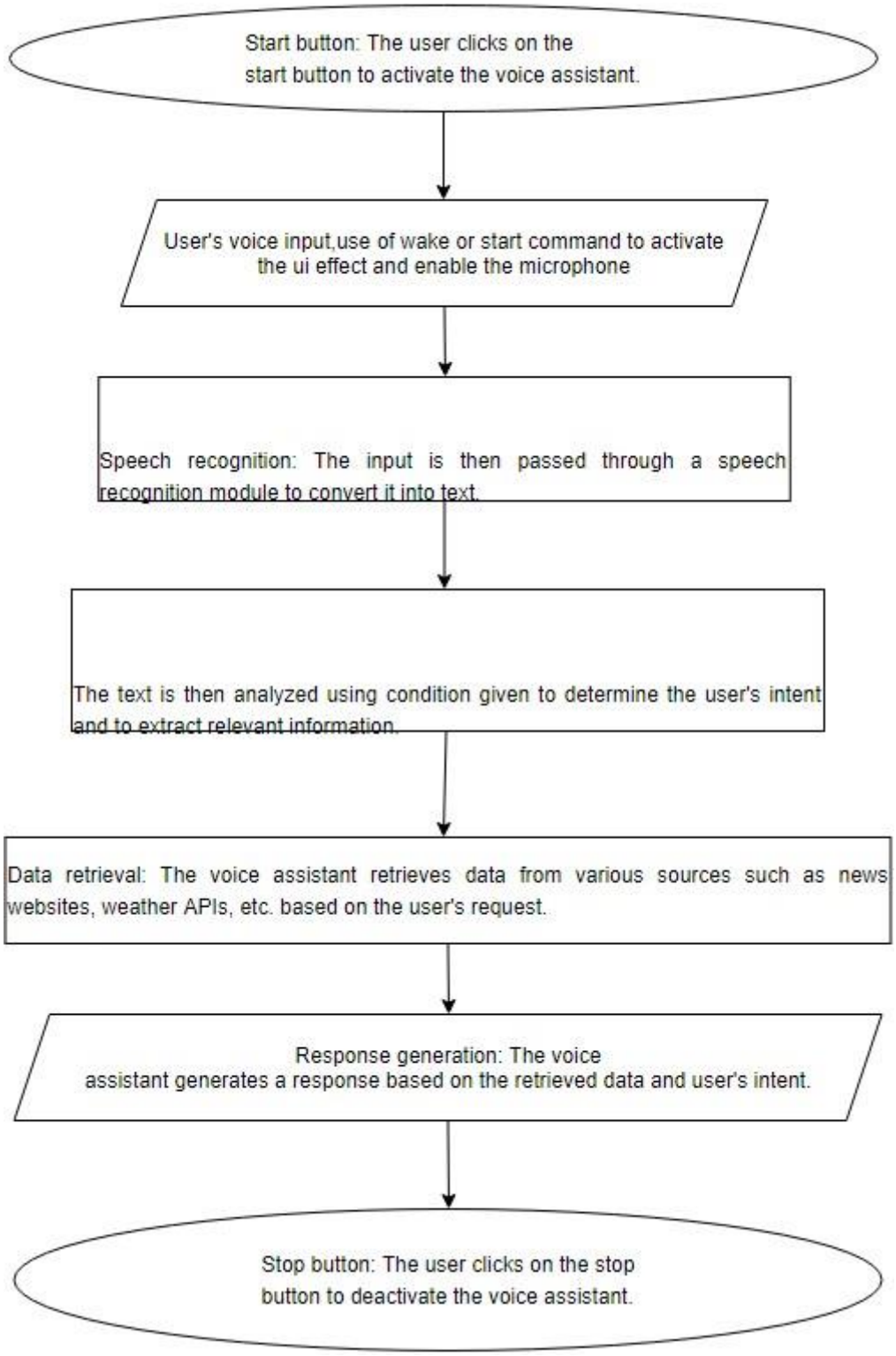
- Connect the Start and Stop buttons to functions or methods in your code that control the behavior of the virtual voice assistant. For example, you can use the start() and stop() methods to start and stop the voice assistant, respectively.

- Use the QLabel widget to display the status of the virtual voice assistant based on its current state. For example, you can update the text of the label to "Jarvis is currently active" or "Jarvis is currently inactive" based on whether the voice assistant is running or not.

- In summary, by following the above steps, you can create an interface for a virtual voice assistant named Jarvis with only two buttons (start and stop) and various futuristic UI and SVG components using Qt Designer.

DFD(Data Flow Diagram):-

This DFD explains the entire working of virtual assistant included with the role of interfave in it:-



This dfd explains how the user input the data how it’s processed and output is given by in addition to how the interface is working in the background.

Source code:-

1)JARVIS.py:-

This is our main python source file that contains the source to all the function s of the voice assistant

2)JARVISUi.py:-

This file contains the source to the user interface that we have created for our virtual assistant .

3)Requirements.txt:-

This file contains all the dependencies that are required for the program

4)Ui folder:-

That consists of ui pics that are used in the user interface

5)Image folder:-

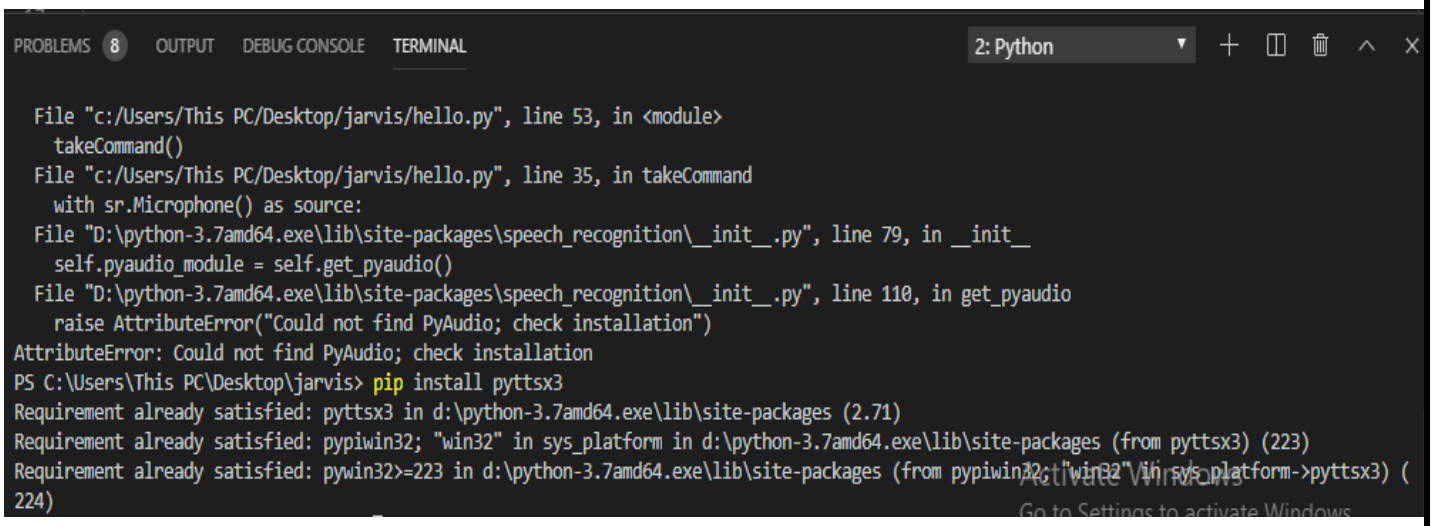
That consists of the background image for our virtual assistant.

Installation of plugins:-

1)pyttsx3:-

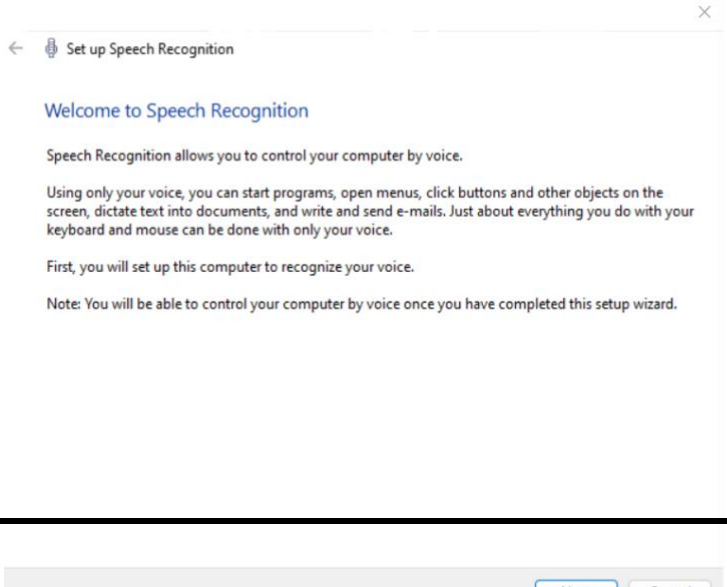
Run this cli command on vsc terminal

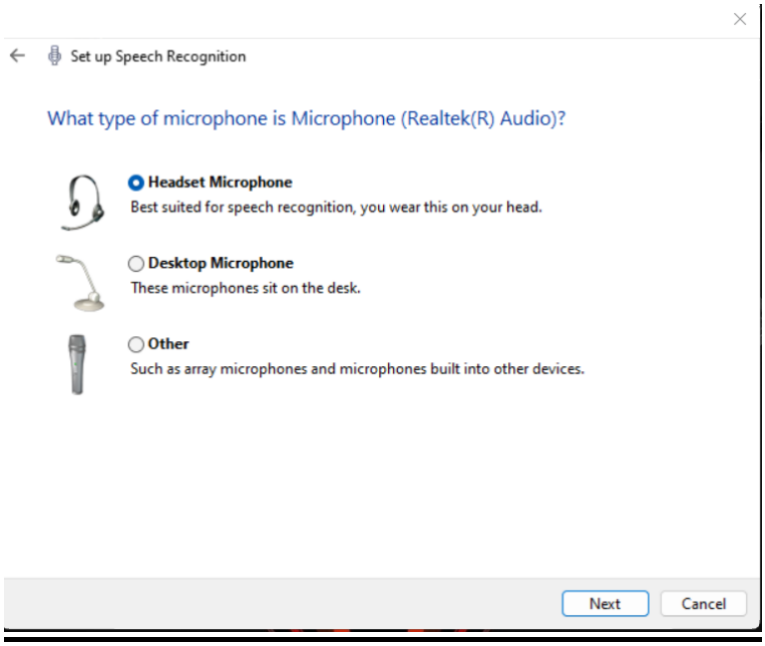
- pip install pyttsx3



2)Sapi:-

Installation in windows and welcome page:-





Installation of plugin:-

```
pip install speechRecognition
```

```
import speechRecognition as sr
```

3)pypiwin32:-

```
pip install pypiwin32.
```

4)pprint;-

```
pip install pprint
```

```
from pprint import pprint
```

5)utils:-

```
pip install pip-utils
```

```
from utils import opening_text
```

6)random2:-

```
pip install random2
```

```
from random import choice
```

7)python-decouple:-

```
pip install python-decouple
```

```
from decouple import config
```

8)os-win:-

```
pip install os-win
```

```
import os
```

9)requests:-

```
pip install requests
```

```
import requests
```

10)PyQt5:-

```
from PyQt5.QtCore import QTimer,QTime,QDate,Qt
from PyQt5.QtGui import QMovie
from PyQt5.QtGui import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
from PyQt5.uic import loadUiType
```

```
pip install PyQt5
```

11)PyPDF2:-

```
import PyPDF2
```

```
pip install PyPDF2
```

12)pytube:-

```
from pytube import YouTube
```

```
pip install pytube
```

Explanation of various modules used in the project with their respective versions:-

The explanation of each of the modules used in the project and their explanation:-

1) SpeechRecognition==3.8.1: A library for performing speech recognition using Google's Speech Recognition API.

2) beautifulsoup4==4.9.3: A library for web scraping, used for parsing HTML and XML documents.

3) psutil==5.8.0: A library for retrieving information about system resources, such as CPU usage, memory usage, and disk usage.

4) PyAutoGUI==0.9.52: A library for automating keyboard and mouse actions, used for automating GUI interactions.

5) pyjokes==0.6.0: A library for generating random jokes, used for adding some humor to the virtual assistant.

6) PyQt5==5.15.2: A set of Python bindings for the Qt application framework, used for creating graphical user interfaces.

7) PyQt5-Qt5==5.15.2: A PyQt5 dependency package containing the Qt5 libraries.

8) pyqt5-plugins==5.15.2.2.0.1: A PyQt5 dependency package containing various plugins for the Qt5 application framework.

9) PyQt5-sip==12.8.1: A PyQt5 dependency package containing the SIP bindings generator for Python.

10) pyqt5-tools==5.15.2.3: A PyQt5 dependency package containing various development tools for the Qt5 application framework.

11) pyttsx3==2.90: A library for text-to-speech conversion, used for speaking responses to the user.

12) pywhatkit==4.7: A library for automating several tasks, such as sending emails, playing songs on YouTube, etc.

13) requests==2.25.1: A library for making HTTP requests, used for fetching data from web APIs.

14) urllib3==1.26.6: A library for handling HTTP requests and responses, used for fetching data from web APIs.

15) wikipedia==1.4.0: A library for interacting with Wikipedia, used for fetching information from Wikipedia pages.

16)DateTime==4.3: A library for working with dates and times, used for fetching and displaying current date and time.

17) opencv-python==4.5.2.52: A library for computer vision, used for processing images and videos.

18) cv2-tools==2.4.0: A library for computer vision, containing several tools for image and video processing.

19) secure-smtplib==0.1.1: A library for sending email messages over SMTP with TLS.

20) random2==1.0.1: A library for generating random numbers.

21) instaloader==4.7.1: A library for downloading pictures and videos from Instagram.

22) PyPDF2>=1.27.5: A library for working with PDF files, used for reading and manipulating PDF files.

23) bs4==0.0.1: A library for web scraping, used for parsing HTML and XML documents.

24) pywikihow==0.5.7: A library for fetching how-to articles from wikiHow.

25) speedtest-cli==2.1.3: A command-line interface for testing internet speed.

26)pytube==10.8.1: A library for downloading YouTube videos.

27) os-sys==2.1.4: A library for getting system information, used for retrieving the operating system's name and version.

28) subprocess32==3.5.4: A backport of the Python 3.2 subprocess module, used for running subprocesses.

29) qt5-applications: A dependency package containing the Qt5 applications

30) qt5-tools: This module provides various tools to work with the PyQt5 GUI framework. PyQt5 is a Python binding of the popular Qt GUI toolkit. Qt5-tools include a set of tools that are used to build and develop graphical user interfaces (GUIs) for desktop applications. It includes tools such as Qt Designer, a graphical user interface (GUI) design tool, and Qt Linguist, a tool for translating application user interfaces.

31)Pillow>=9.0.1: Pillow is a Python Imaging Library (PIL) fork that adds support for opening, manipulating, and saving many different image file formats. Pillow provides an easy-to-use set of classes and functions for working with images in Python. It is widely used for image processing and computer vision applications. Pillow>=9.0.1 specifies that the code requires Pillow version 9.0.1 or later.

32)PyWave==0.5.0: PyWave is a Python module that provides a set of functions for reading, writing, and processing audio waveforms. It supports a wide range of audio formats and provides an easy-to-use interface for working with audio data. PyWave==0.5.0 specifies that the code requires PyWave version 0.5.0.

33)phonenumbers==8.12.23: phonenumbers is a Python module that provides functions for working with phone numbers. It can parse, format, and validate phone numbers for different countries and regions. phonenumbers==8.12.23 specifies that the code requires phonenumbers version 8.12.23.

34)folium==0.12.1: folium is a Python module that provides an easy-to-use interface for creating interactive maps and visualizations using Leaflet.js. It can be used to create maps with different types of markers, heatmaps, choropleth maps, and more. folium==0.12.1 specifies that the code requires folium version 0.12.1.

35)opencage==1.2.2: opencage is a Python module that provides a geocoding API to convert latitude and longitude coordinates to addresses and vice versa. It supports multiple languages and provides accurate results. opencage==1.2.2 specifies that the code requires opencage version 1.2.2.

36)ffmpeg-normalize==1.22.1: ffmpeg-normalize is a Python module that provides a command-line tool for normalizing audio and video files using the ffmpeg library. It can be used to adjust the volume, apply filters, and convert formats. ffmpeg-normalize==1.22.1 specifies that the code requires ffmpeg-normalize version 1.22.1.

37)PyAudio==0.2.11: PyAudio is a Python module that provides bindings for the PortAudio library, which allows Python applications to record and play audio streams. It supports multiple platforms and provides an easy-to-use interface for working with audio data. PyAudio==0.2.11 specifies that the code requires PyAudio version 0.2.11.

These all can be downloaded from pypi.org or directly with the help of cli command using pip command

References(According to IEEE References Citation):-

- Sharma, Shubham. “Research.” Geeksforgeeks, shrey, 30 Apr. 2023, •www.geeksforgeeks.org/python-text-to-speech-by-using-pyttsx3/. Accessed 30 Apr. 2023. for_research_purpose.

- ---. “Sapi5.” *Learn.microsoft.com*, Shubham Sharma, 30 Apr. 2023, learn.microsoft.com/en-us/previous- versions/windows /desktop/ms723627(v=vs.85). Accessed 4 May 2023.

- Sharma, Shubham. “Python_concepts.” Www.python.org, Shubham Sharma, 30 Apr. 2023, •www.python.org/shell/. Accessed 4 May 2023.

- Sharma, Shubham. “DFD_Concepts.” Www.geeksforgeeks.org, Shubham Sharma, 30 Apr. 2023, • www.geeksforgeeks.org/what-is-dfddata-flow-diagram/. Accessed 4 May 2023.

- ---. “Pypi_Modules_Info.” Pypi.org, Shubham Sharma, 30 Apr. 2023, •pypi.org/project/ctypes/. Accessed 4 May 2023.

- ---. “UI_Components.” Gifer.com/, Shubham Sharma, 30 Apr. 2023, •gifer.com/en/gifs/jarvis. Accessed 4 May 2023.

- ---. “UI_Components_2.” Giphy.com, Shubham Sharma, 30 Apr. 2023, •giphy.com/gifs/11nCYa0XqPKa2s. Accessed 4 May 2023.

- ---. “UI_Loading_Component.” Loading.io, Shubham Sharma, 30 Apr. 2023, •loading.io/animation/text/. Accessed 4 May 2023.

- ---. “Query.” Stackoverflow.com, Shubham Sharma, 30 Apr. 2023, •stackoverflow.com/. Accessed 4 May 2023.

- ---. “Project_Research.” Www.freecodecamp.org, Shubham Sharma, 30 Apr. 2023, www.freecodecamp.org/. Accessed 4 May 2023.

Advantages:-

1.Hands-free operation: One of the main advantages of voice virtual assistants is that users can operate them without using their hands. This feature is especially useful while driving or performing other activities that require both hands.

2.Accessibility: Voice virtual assistants can be used by people with disabilities, such as those who are visually impaired or have mobility issues, as they do not require any physical interaction with the device.

3.Convenience: With a voice virtual assistant, users can get information or perform tasks quickly and easily without having to type or navigate through menus.

4.Personalization: Voice virtual assistants can learn about users' preferences and habits over time and

5.personalize their responses accordingly, providing a more customized experience.

6.Integration: Many voice virtual assistants can be integrated with other smart home devices, allowing users to control their lights, thermostat, or security system with their voice.