

ABSTRACT

This project includes an implementation of an intelligent voice assistant for desktop. Where functionality on current existing applications on other platforms is compared. Until this day, there has not been any good alternative for Windows, so this project aims to implement a voice assistant for the Windows platform while describing the difficulties and challenges that lies in this task.

Voice assistants are software agents that can interpret human speech and respond via synthesized voices. Apple's Siri, Amazon's Alexa, Microsoft's Cortana, and Google's Assistant are the most popular voice assistants and are embedded in smartphones or dedicated home speakers. Users can ask their assistants questions, control home automation devices and media playback via voice, and manage other basic tasks such as to-do lists, checking weather, Google searching engine, Wikipedia searching engine, camera, and many more functionalities with verbal commands. This column will explore the basic workings and common features of today's voice assistants. It will also discuss some of the privacy and security issues inherent to voice assistants and some potential future uses for these devices.

The attempt has been made to develop an "Intelligent Personal Voice Assistant using Python" which helps people to control device with their voice (speech), extract information and perform tasks on their desktop.

INDEX

CHAPTER NO	TOPICS	PAGE . NO
CHAPTER 1	INTRODUCTION	08
CHAPTER 2	PROBLEM STATEMENT	11
CHAPTER 3	SYSTEM ANALYSIS	13
CHAPTER 4	IMPLEMENTATION	20
CHAPTER 5	CODING IN THE PROJECT	28
CHAPTER 6	SNAPSHOTS	53
CHAPTER 7	TESTING	58
CHAPTER 8	FUTURE SCOPE	63
CHAPTER 9	CONCLUSION	65
CHAPTER 10	BIBLIOGRAPHY AND REFERENCE	67

INTRODUCTION

In today's era almost all tasks are digitalized. We have Smartphone in hands and it is nothing less than having world at your fingertips. These days we are not even using fingers. We just speak about the task and it is done. There exist systems where we can say Text Dad, "I'll be late today." And the text is sent. That is the task of a Virtual Voice Assistant.

This system is designed to be used efficiently on desktops. Personal assistant software improves user productivity by managing routine tasks of the user and by providing information from online sources to the user. People always wanted to talk to computers almost from the moment the first computer was invented. Just a few decades ago, the idea of holding meaningful conversation with a computer seemed futuristic, but the technology to make voice interfaces useful and widely available is already here. Several consumer-level products developed in the last few years have brought inexpensive voice assistants into everyday use, and more features and platforms are being added all the time.

Users can do everything from asking simple informational questions to playing music and dialing their phone or turning lights on and off via voice control. This column will explore the basic workings and common features of today's voice assistants. It will also discuss some potential future uses for these devices.

Voice assistants comes in small packages and can perform a variety of task and actions after hearing a wake up word or command. They can play music, turn on the lights, they can answer any question and also place an order for you.

This project was started on the premise that there is sufficient amount of openly available data and information on the web that can be utilized to build a virtual assistant that has access to making intelligent decisions for routine user activities.

1.1 Objective of Project

Main objective of building personal assistant software (a virtual assistant) is using semantic data sources available on the web, user generated content and providing knowledge from knowledge databases. The main purpose of an intelligent virtual assistant is to answer questions that users may have.

This project is focusing on the Desktop App development over the voice control (recognition, generate and analyze corresponding commands, intelligent responses automatically), Google products and relevant APIs (Google weather, Google search and etc), Wikipedia API and desktop device references ranging from Speech-To-Text, Text-To-Speech technology,

Virtual assistants can tremendously save you time. We spend hours in online research and then making the report in our terms of understanding. David can do that for you. Provide a topic for research and continue with your tasks while David does the research. Another difficult task is to remember test dates, birthdates or anniversaries. It comes with a surprise when you enter the class and realize it is class test today. Just tell David in advance about your tests and he reminds you well in advance so you can prepare for the test.

1.2 Limitations of Project

- They depend on an Internet connection.
- Voice assistants use single commands. For now, these consist mostly of fixed phrases. Effectively, they push one button or set one dial.
- As more flexible natural language understanding technology is becoming available, interpretations of speech commands may become ambiguous. With commands resulting into actions, misunderstandings can be risky.
- Voice assistants cannot integrate context data, such as what I'm currently working on.

These shortcomings limit voice assistants from elevating their status from transactionary to really helpful. They need a semantic level of interaction to support more complex activities.

PROBLEM STATEMENT

Usually, user needs to manually manage multiple sets of applications to complete one task. For example, a user trying to check for currently active covid cases for a particular country. In order to achieve this user needs to open a browser and search for a website which displays this particular data and then user needs to enter the country name there then finally user will get all the data which the user is looking for. So there is a need of a system that can manage tasks effortlessly.

There are number of people who have issues in voice recognition. These systems can understand English phrases but they fail to recognize in our accent. Our way of pronunciation is way distinct from theirs. Also, they are easy to use on mobile devices than desktop systems.

There is need of a virtual assistant that can understand English in Indian accent and work on desktop system. When a virtual assistant is not able to answer questions accurately, it's because it lacks the proper context or doesn't understand the intent of the question. Its ability to answer questions relevantly only happens with rigorous optimization, involving both humans and machine learning. Continuously ensuring solid quality control strategies will also help manage the risk of the virtual assistant learning undesired bad behaviors. They require large amount of information to be fed in order to work efficiently.

Virtual assistant should be able to model complex task dependencies and use these models to recommend optimized plans for the user. It needs to be tested for finding optimum paths when a task has multiple sub-tasks and each sub-task can have its own sub-tasks. In such a case here can be multiple solutions to paths, and it should be able to consider user preferences, other active tasks, and priorities in order to recommend a particular plan.

SYSTEM ANALYSIS

Personal assistant software is required to act as an interface into the digital world by understanding user requests or commands and then translating into actions or recommendations based on agent's understanding of the world.

David focuses on relieving voice as primary means of user input. Agent then applies voice recognition algorithms to this input and records the input. It then use this input to call one of the personal information management applications such as task list or calendar to record a new entry or to search about it on search engines like Google. Focus is on capturing the user input through voice, recognizing the input and then executing the tasks if the agent understands the task. Software takes this input in natural language, and so makes it easier for the user to input what he or she desires to be done.

3.1 FEASIBILITY STUDY

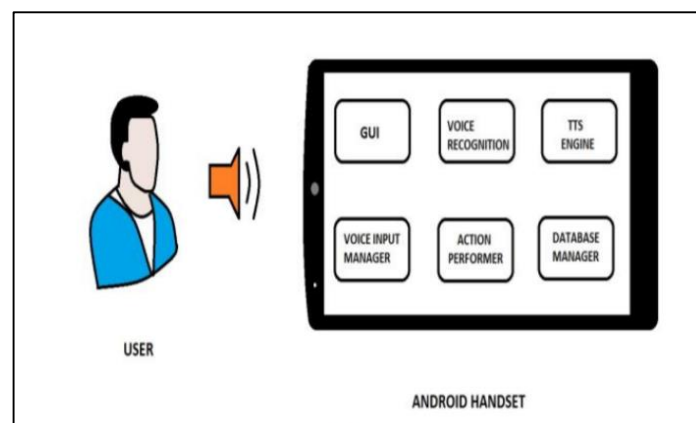
Feasibility study can help you determine whether or not you should proceed with your project. It is essential to evaluate cost and benefit. It is essential to evaluate cost and benefit of the proposed system. Five types of feasibility study are taken into consideration.

- 1. Technical feasibility:** It includes finding out technologies for the project, both hardware and software. For virtual assistant, user must have microphone to convey their message and a speaker to listen when system speaks. These are very cheap now a days and everyone generally possess them. Besides, system needs internet connection. While using David, make sure you have a steady internet connection. This should not be an issue in this era where almost every home and office has Wi-Fi.
- 2. Operational feasibility:** It is the ease and simplicity of operation of proposed system. System does not require any special skill set for users to operate it. In fact, it is designed to be used by almost everyone. Kids who still don't know to write can read out problems for system and get answers.
- 3. Economical feasibility:** Here, we find the total cost and benefit of the proposed system over current system. For this project, the main cost is documentation cost. As far as maintenance is concerned, David won't cost too much.

4. **Organizational feasibility:** This shows the management and organizational structure of the project. This project is built by a team of two members. The management tasks are all carried out by the team. That won't create any management issues and will increase the feasibility of the project.
5. **Cultural feasibility:** It deals with compatibility of the project with cultural environment. Virtual assistant is built in accordance with the general culture. This is the reason why we named it as David.

3.2 SYSTEM DESIGN

Proposed System: The following figure gives a brief idea about the system architecture. The following figure gives a brief idea about the proposed system.



3.2.1 Main Modules:

Speech Recognition module: The system uses Google's speech recognition system for converting speech input to text. This module uses the library of speechRecognition 3.8.1 for performing speech recognition, with support for several engines and APIs, online and offline.

Python backend: To get the output from the speech recognition module and then identifies whether the command or the speech output is an API Call, Context Extraction, and System Call then the output is send back to the python backend to give the required output to the user.

API Calls: It allows two applications to talk to each other. API is working as a messenger that delivers request to the provider that are requesting it from and then delivers the response back.

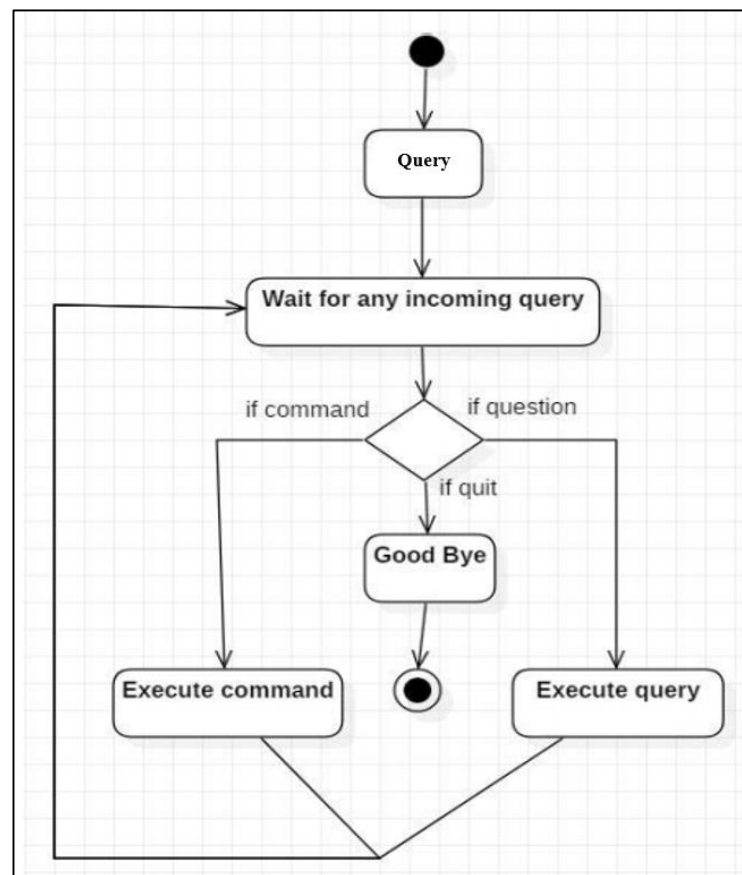
Context Extraction: It is the function of automatically or robotic extracting structured information from unstructured or semi-structured or both machine readable documents. In majority of the instance this task concerns processing of human language texts by the method of natural language processing (NLP).

System Calls: It is the programmatic method in which a computer program (function) appeal a service from the kernel of the operating system it is executed on. It includes hardware related assistant and services like access of hard disk drive, creation and execution of new processes, and interacting with integral kernel services like process scheduling. It delivers an essential interface between a process and the operating system.

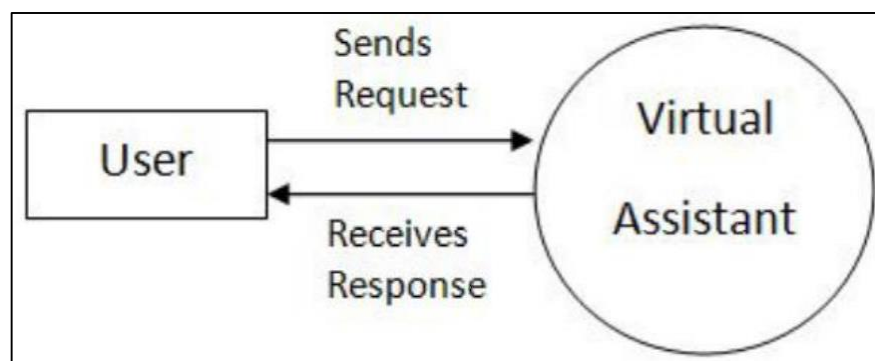
Text-To-Speech (TTS) Engine: A text-to-speech (TTS) system converts normal language text into speech. Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database. The output is given in the form of speech

GUI: It is used to interact with the user. GUI reflects the basic appearance of the application.

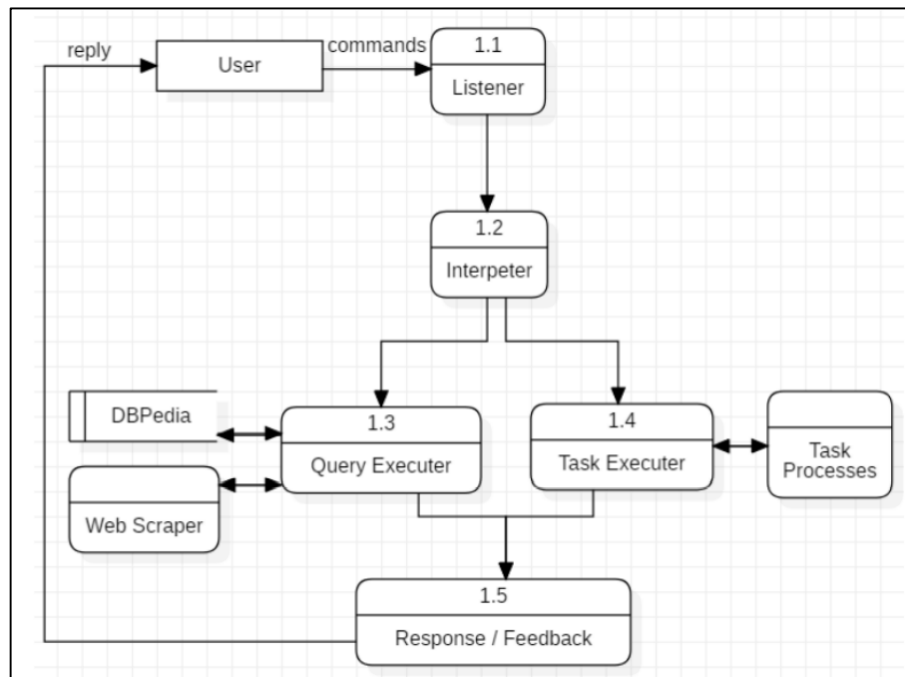
3.3 Flow Diagram:



DFD Level 0:



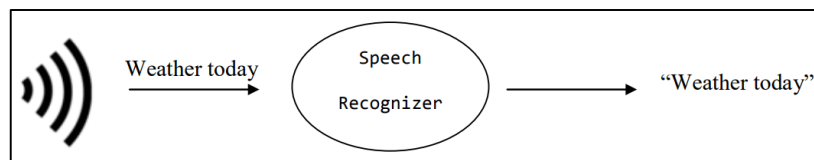
DFD Level 1:



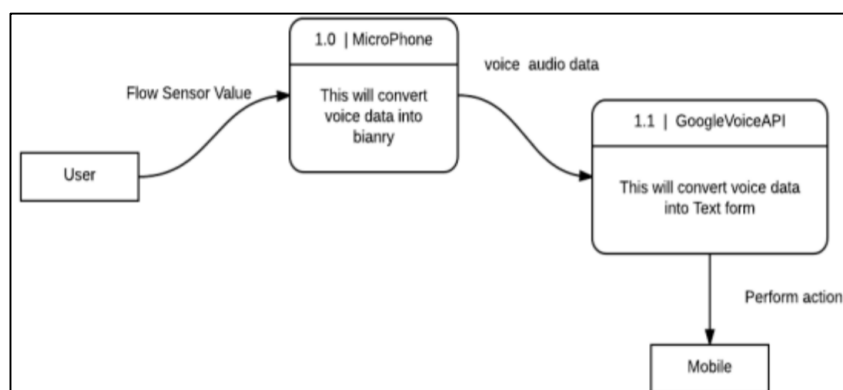
Speech Recognition DFD:

User gives the input into the form of voice, this voice command is recognized by the application. Then action is performed as per the command given. Command given is compared with the database.

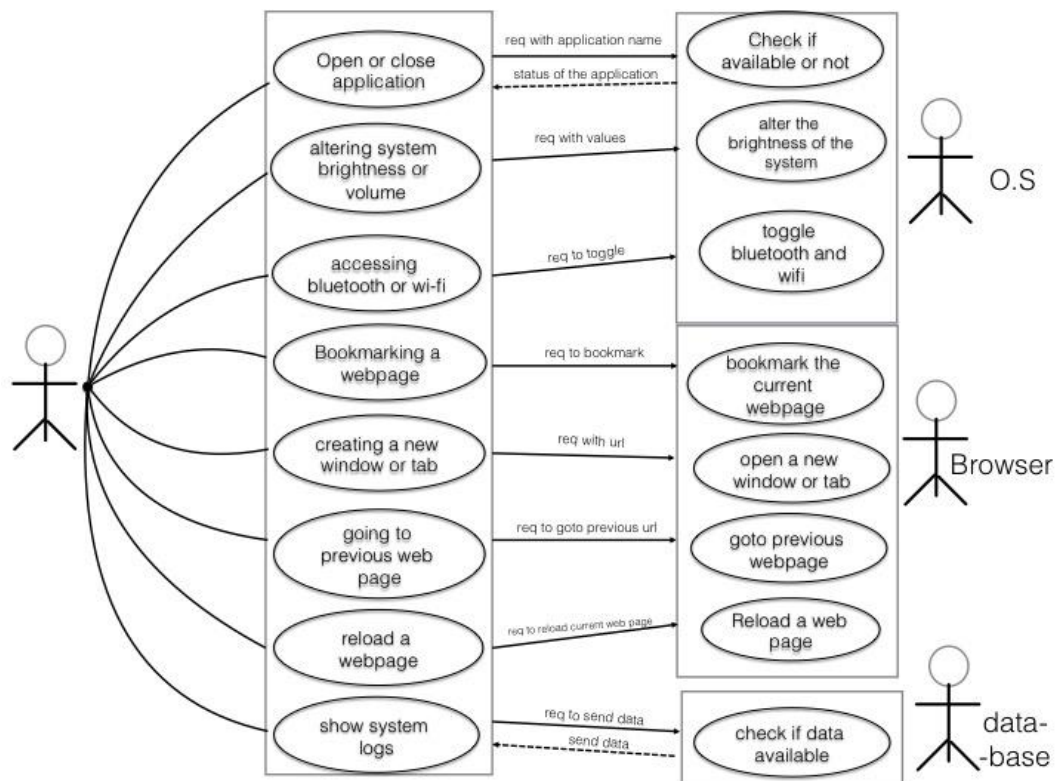
Level 0:



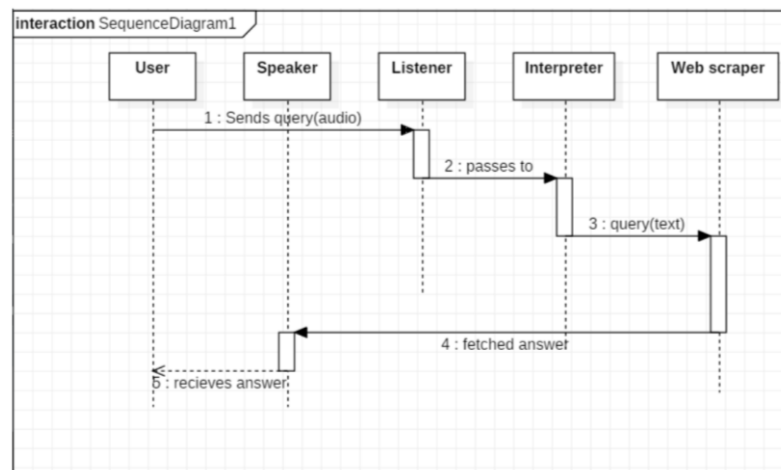
Level 1:



Use Case Diagram:



Sequence Diagram:



The above sequence diagram shows how an answer asked by the user is being fetched from internet. The audio query is interpreted and sent to Web scraper. The web scraper searches and finds the answer. It is then sent back to speaker, where it speaks the answer to user.

IMPLEMENTATION

The project aim is to build a personal voice assistant that will make easy for users to use computer with voice command and make task easier. To implement intelligent personal voice assistant, python libraries and Google Speech Recognition API's are used for speechRecognition module and to interpret voice response.

Intelligent personal voice assistants was meant to control the applications online as well as offline and to analyze the performance of the project the working of Intelligent personal voice assistant with desktop application can be opening YouTube with voice command.

The program should firstly be started on the desktop; the initial mode of the program is Voice mode since this program aims at making a voice assistant program. After the program has been started, the user should have correct voice input "command/request" to make those functions work properly.

Input is given by user in the form of voice. Using microphone, voice is converted in binary. GoogleVoiceAPI will convert this voice data in text form and then the action is performed according to the command given by the user by comparing with the match case.

And this program includes the functions and services of: calling services, text message transformation, event handler, services, music player service, checking weather, Google searching engine, Wikipedia searching engine, robot chat, and camera.

Once the user gives a command to the assistant the program first identifies the query and then it converts the query into string after that the assistant processes the query based upon the match cases provided. For example if the user asks to "open Notepad" the main part here is notepad which is an application program and the attached keyword is open thus the match case for the query will be "Notepad" and the operation associated with the notepad which is "open". Finally the assistant initiates a system call to launch a process for opening notepad and the Notepad opens, and assistant gives an acknowledgement to the user that the notepad has been opened.

The details below explain how all functions work and different possibilities while facing different commands.

Calling service: The calling function allows the users to give a call to the person in the contacts. By giving a correct command with the calling request to a stored person, the Assistant will search for the contact name and get the contact of the person, then successfully placing the call the person.

Text Message The calling function allows the users to send a whatsapp message to the person in the contacts. By giving a correct command contains the text or message keyword to send message together with the destination person; the program will navigate to the sending message function on the whatsapp app, message content. The message will be sent to the destination immediately if the user selects to send it with the correct content.

Note taking: The application allows the user to set as many events as they want. Customers set the events with the content and note name, the program switch to the note taking interface with the content and the title, and the event will be stored immediately.

Music player: The music player offers the services to the user to play a named or random song on YouTube depending on the request. The music player service will play the specific song according to the name given by the user. By giving the correct commands, the working music player can stop playing

Checking weather: Weather service provides the user the weather condition in different city. This service works in the same logic and gives back different result depending on the requested city. The weather service return the weather condition of the city with the humidity, wind speed, temperature scope and display in a formalized entity which can be easily read by the user.

Google searching engine: The search engine enable the use to search anything on Google. By detecting the search keyword and search request, the Google search engine will returns the result list displayed to the user through the UI.

Wikipedia searching engine: The search engine enable the use to search anything on Wikipedia. By detecting the Wikipedia keyword and search request, the Wikipedia search engine will returns the Wikipedia result displayed to the user.

General Chat: The assistant exchanges dialog with the user like a normal human enables the user to talk with the Assistant to have fun.

Camera: The camera function enables the user to capture the current view with the camera on the desktop.

Computational problems: The use of certain API's enable the user to ask the assistant for complex solution like solving math operation and asking knowledge based questions like "Who is the prime minister of India " the assistant is smart enough to answer all such questions with accuracy.

Modifying system settings: The assistant is able to change or modify system setting as per user command like changing the brightness level and volume level.

Project Modules:

- **Scraping:**

1. **BeautifulSoup:** BeautifulSoup is a library that makes it easy to scrape information from web pages. It sits atop an HTML or XML parser, providing Pythonic idioms for iterating, searching, and modifying the parse tree.
2. **Wikipedia:** Wikipedia is a Python library that makes it easy to access and parse data from Wikipedia. Search Wikipedia, get article summaries, get data like links and images from a page, and more. Wikipedia wraps the MediaWiki API so you can focus on using Wikipedia data, not getting it.

- **System modules:**

1. **OS:** The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality. The `*os*` and `*os.path*` modules include many functions to interact with the file system.
2. **Psutil:** It is a Python cross-platform library used to access system details and process utilities. It is used to keep track of various resources utilization in the system. Usage of resources like CPU, memory, disks, network, sensors can be monitored
3. **Pytttsx3:** It 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 `pytttsx3.init()` factory function to get a reference to a `pytttsx3`. Engine instance. it is a very easy to use tool which converts the entered text into speech. The `pytttsx3` module supports two voices first is female and the second is male which is provided by "sapi5" for windows.
4. **Speech_recognition:** Speech Recognition is an important feature in several applications used such as home automation, artificial intelligence, etc. This article aims to provide an introduction on how to make use of the `SpeechRecognition` library

of Python. This is useful as it can be used on microcontrollers such as Raspberri Pis with the help of an external microphone.

5. **Tkinter:** Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.
6. **Platform:** The Platform module is used to retrieve as much possible information about the platform on which the program is being currently executed. Now by platform info, it means information about the device, it's OS, node, OS version, Python version, etc.
7. **Time:** This module provides various time-related functions. For related functionality, see also the datetime and calendar modules. time module available in Python which provides functions for working with times, and for converting between representations. The function *time.time()* returns the current system time.

- **Installed modules:**

1. **pywhatkit:** PyWhatKit is a Python library for Sending whatsapp message at certain time, it has several other features like Send WhatsApp messages, Play a YouTube video, Perform a Google Search and Get information on particular topic.
2. **covid:** A new Python library which tells the COVID-19 related information (country-wise) and it show that how many cases of confirmed, active, deaths, recovered found in that particular Country.
3. **webbrowser:** webbrowser module provides a high-level interface which allows displaying Web-based documents to users. The webbrowser module can be used to launch a browser in a platform-independent manner. The webbrowser module provides a high-level interface to allow displaying Web-based documents to users.
4. **pyautogui:** PyAutoGUI lets Python control the mouse and keyboard, and other GUI automation tasks. PyAutoGUI is a cross-platform GUI automation Python module for human beings. Used to programmatically control the mouse & keyboard.
5. **speedtest:** Speedtest-cli is a module that is used in the command-line interface for testing internet bandwidth using speedtest.net.
6. **PIL:** The Python Imaging Library adds image processing capabilities to your Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities. The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

- **API's**

1. **WolframAlpha:** The WolframAlpha Web service API provides a web-based API allowing the computational and presentation capabilities of Wolfram Alpha to be integrated into web, mobile, desktop, and enterprise applications. Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledgebase and AI technology.

- **User defined**

1. **generalResponse:** This module helps the assistant to answer user's general queries "like how are you" etc. The module contains a dictionary with user query as keys and assistant response as value to the key.
2. **game:** This module is a Tic tac toe module which will be used when user queries for playing game.
3. **YoututbeDownloader:** This module is for downloading YouTube videos based upon the provided link by the user. This is an extension to a module named pytube.

Some Usage Commands:

1. "Play a song for me", the program will ask which song you want to play.
2. "I want to call", make a phone call to Lucy. The program will capture the command keyword "call" and then it will ask for the contact name.
3. "Send a message", the program will capture the keyword "message" and then it will ask for the name of the contact along with the text you want to send.
4. "Set up a meeting at 10", the Program will first capture the keyword "meeting" then the event activity will be start with the add event dialog, automatically creating a meeting link.
5. "Google Spain", the keyword 'Google' is detected and the result will be presented on the web browser by searching 'Spain' on Google.
6. "Open the camera", as the keyword 'open camera' is detected, the camera is started. And the program will take a photo within 5 seconds.

HARDWARE AND SOFTWARE REQUIREMENTS

The software is designed to be light-weighted so that it doesn't much resources of the machine and eventually be a burden on the machine running it. This system is being build keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirement for virtual assistant.

RAM 512MB or more. Software: • Windows 7(32-bit) or above. • Python 2.7 or later • Chrome Driver • Selenium Web Automation • SQLite

Software Requirements:

- Windows 7(32-bit) or above.
- Python 3.9.1 or later
- Pycharm Community Edition 2020.3

Hardware Requirements:

- Ram: 512Mb minimum, 1GB Recommend.
- Storage: 2GB Recommend.
- CPU: Core 2 Duo Minimum, I3 5th Gen Recommend

Functional Requirements:

The main purpose of functional requirements within the requirement specification document is to define all the activities or operations that take place in the system. These are derived through interactions with the users of the system.

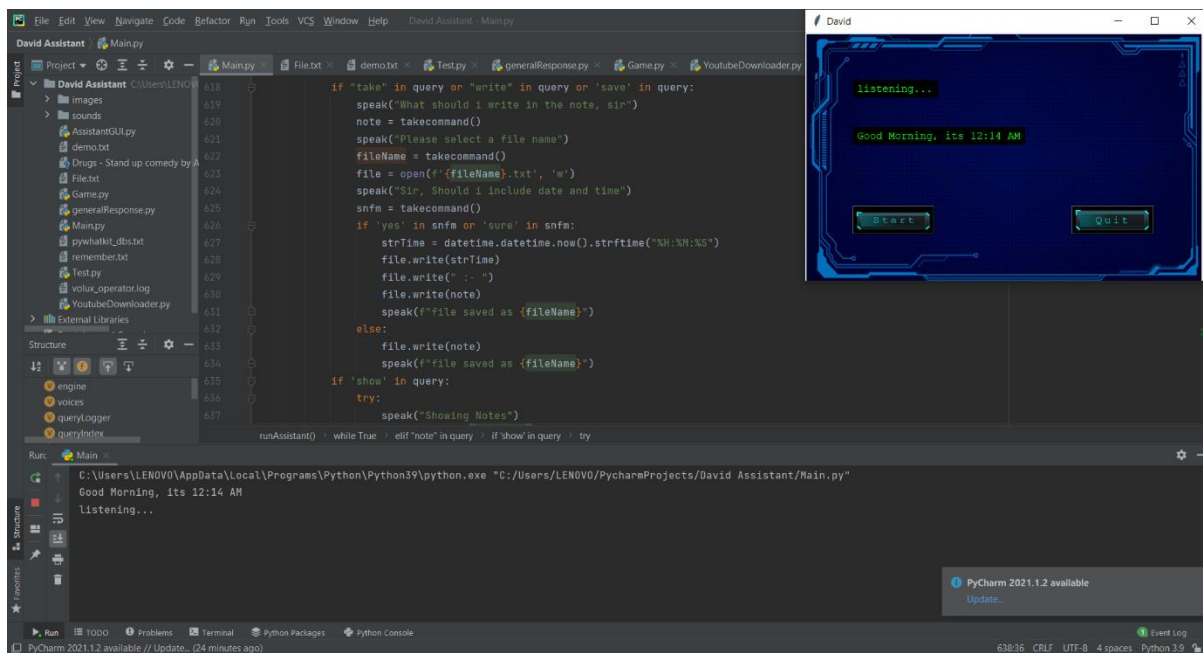
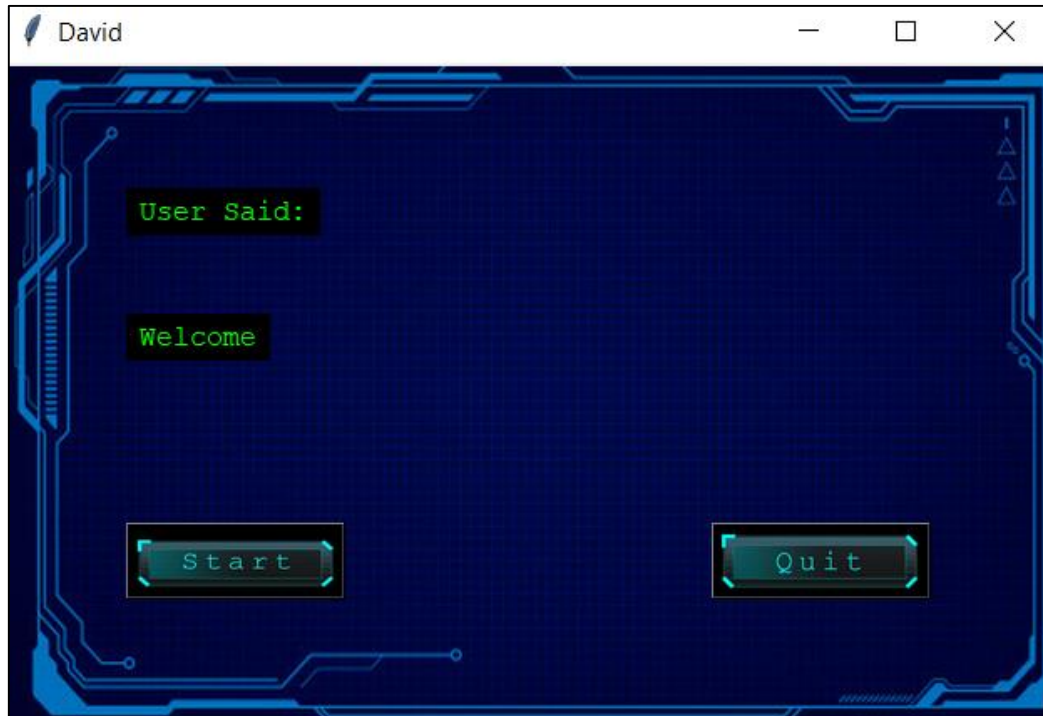
1. Accepting an audio signal,
2. Deciding whether the audio signal is addressed to the software agent,
3. Attempting to map a relevant signal onto a word sequence,
4. Determining the rover action request associated with the understood word sequence,
5. Providing appropriate feedback to the user.

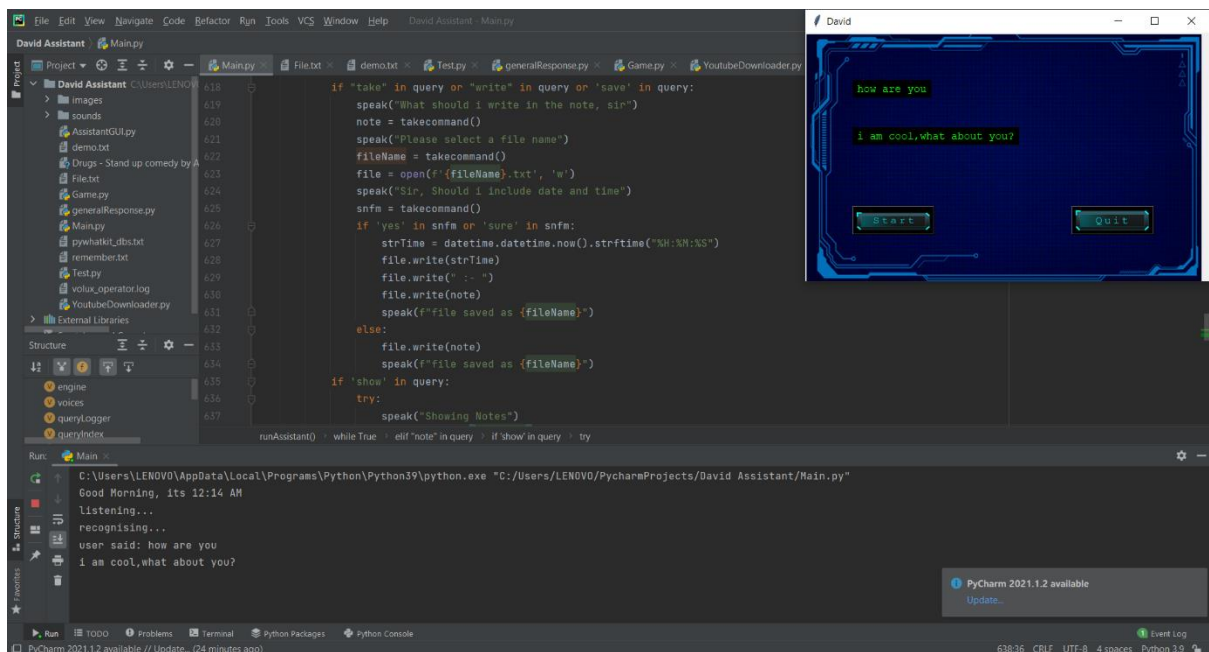
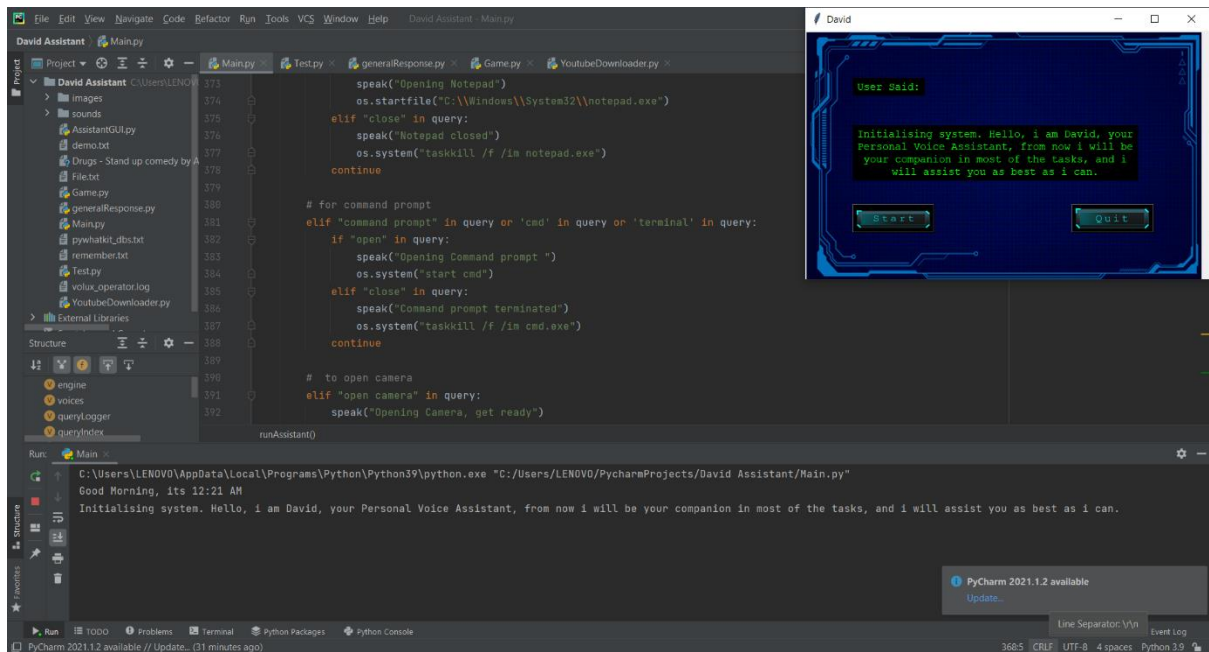
Non-functional Requirements:

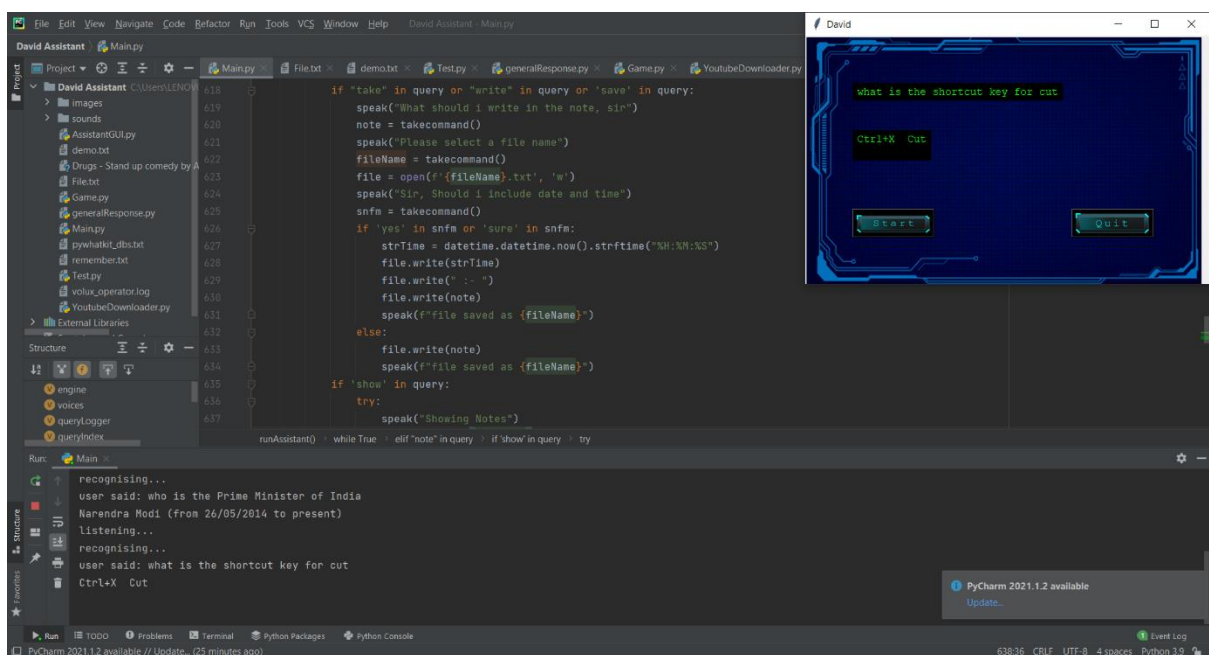
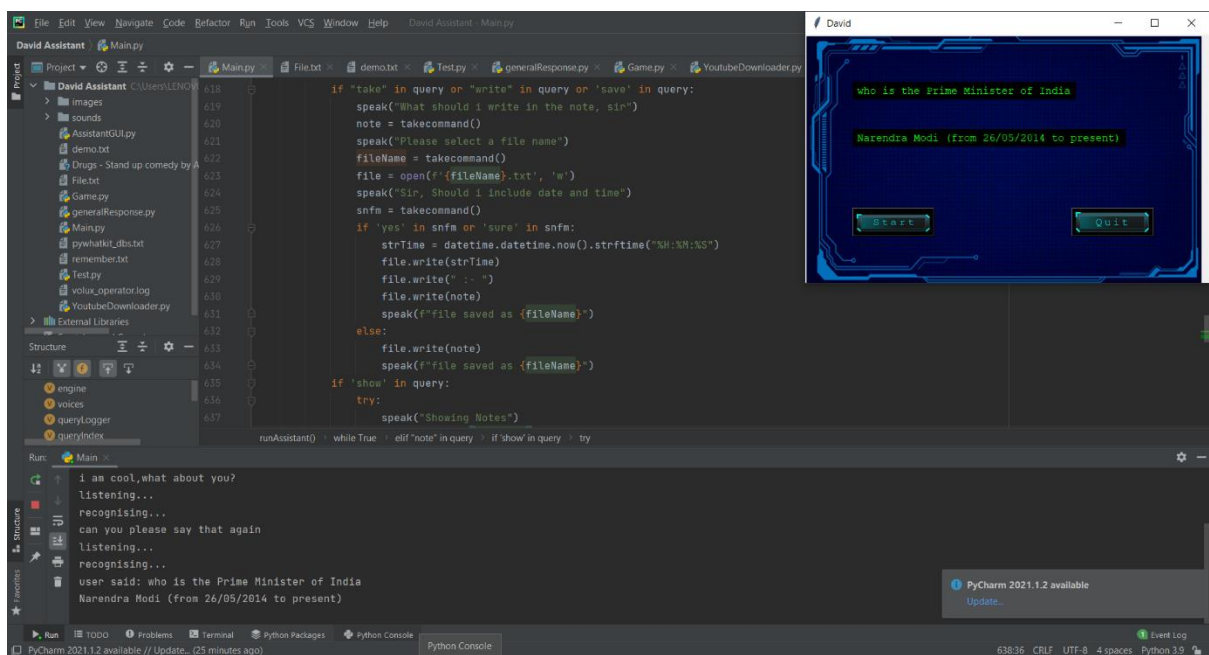
Non-functional requirements describe user-visible aspects of the system that are not directly related to functionality of the system.

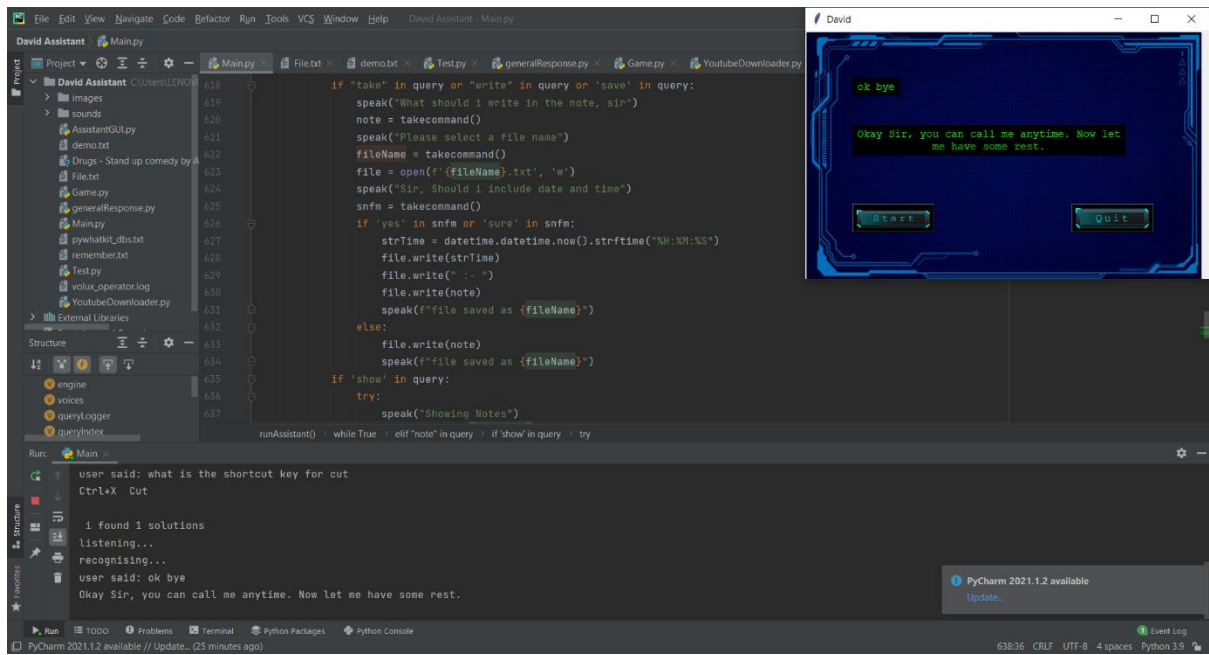
- The system ensures safety, security and usability, which are observable during operation (at run time).
- The system is adaptable to different situations.
- The project has good and compact UI using Tkinter with responsive interface.
- The project is light on resources.

SNAPSHOTS









TESTING FORMS AND REPORTS

7.1 Introduction to Testing

It is the stage of implementation, which ensures that system works accurately and effectively before the live operation Commences. It is a confirmation that all are correct and opportunity to show the users that the system must be tested with text data and show that the system will operate successfully and produce expected results under expected conditions.

Before implementation, the proposed system must be tested with raw data to ensure that the modules of the system work correctly and satisfactorily. The system must be tested with valid data to achieve its objective.

The purpose of system testing is to identify and correct errors in the candidate system. As important as this phase is, it is one that is frequently compromised. Typically, the project the schedule or the user is eager to go directly to conversion. Actually, testing is done to achieve the system goal. Testing is vital to the parts of the system are correct; the goal will be successfully achieved. Inadequate testing or non-testing leads to errors that may not appear until months later.

Types of Testing performed on the System:

1. Unit Testing

It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.

Performed tests:

1. This type of testing was performed mainly on singular functionalities such as sending message and calling. The testing was done thoroughly on this functionalites the main objective was to reduce the response time in sending message and recieving acknowledgement from the assistant.
2. The covid API was tested multiple times for receiving covid updates in least possible time.
3. The functionality which tests internet connectivity was tested thoroughly in order to minimize the response time while getting the internet speed.
4. The weather API was tested with multiple input data for stressing the API to improve performance.
5. The system was tested for launching computer applications in least possible time was our goal to imporve performance of the system.

We conducted unit testing whenever we worked on a new functionality and tested it thoroughly with all possible test inputs. And it has passed all the test cases.

2. Integration Testing

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output.

After developing a new functionality and testing it thoroughly individually we integrated with the existing system and then performed all the testing operations and methodologies in order to produce expected final output. And it produced the desired result without exceeding time limit.

1. Test Case 1

Test Title: Response Time

Test ID: T1

Test Priority: High

Test Objective: To make sure that the system respond back time is efficient.

Description: Time is very critical in a voice based system. As we are not typing inputs, we are speaking to them. The system must also reply in a moment. User must get instant response of the query made. Otherwise the response timeout functionality should be there.

2. Test Case 2

Test Title: Accuracy

Test ID: T2

Test Priority: High

Test Objective: To assure that answers retrieved by system are accurate as per gathered data.

Description: A virtual assistant system is mainly used to get precise answers to any question asked. Getting answer in a moment is of no use if the answer is not correct. Accuracy is of utmost importance in a virtual assistant system. Systems must produce accurate output in order to provide better service to the user.

3. Test Case 3

Test Title: Approximation

Test ID: t3

Test priority: Moderate

Test Objective: To check approximate answers about calculations. Description: There are times when mathematical calculation requires approximate value. For example, if someone asks for value of PI the system must respond with approximate value and not the accurate value. Getting exact value in such cases is undesirable. If there are multiple answers to a question then the most suitable one should be produced as an output with approximation techniques.

4. Test Case 4

Test Title: Connectivity

Test ID: t4

Test priority: High

Test Objective: To check proper network connectivity to the internet

Description: The system must check for the connectivity issues in order to provide seamless user experience. If connectivity is slow then the response time will also be affected eventually affecting the performance of the system, thus making this one of the most crucial part of the system.

5. Test Case 5

Test Title: Input processing

Test ID: t5

Test priority: High

Test Objective: To check for proper input processing throughout the lifecycle of the input.

Description: The system must be respondent to the user in a limited time with high accuracy for which the processing of the input must be accurate, fast and must be processed to the correct destination, in order to retrieve correct output.

VALIDATION TESTING

The validation testing is performed for all the data in the system. The data are completely validated according to the requirement.

In this testing, software is completely assembled as package, interfacing errors have been uncovered and correction testing begins after each one of the two possible conditions exists.

The function or performance characteristic's is confirm specification and are accepted. A deviation from the specification is uncovered and efficiency list is created.

OUTPUT TESTING

Various outputs have been generated by the system. The system generated output and the desk-calculated values have been compared. All the output is perfect as the company desires. It begins with low volumes of transactions based on live tone. The volume is increased until the maximum level for each transaction type is reached.

The total system is also tested for recovery and fallback, after various major failures to ensure that no data are lost during the emergency time.

Test Results

The VIPA has undergone all the type of testing such as unit testing, integration testing, validation testing and system testing and it is ensured that the information properly flows in and out of the system, and produces the correct expected output as per the user requirement. All the loop holes and error handling paths are tested and found that everything is working well even under extreme conditions.

Maintenance

After the program is completed, the program still needs long term maintenance to make it available and stable to execute. The program will be test after a certain period of time and debug each of the function and possible bugs, whenever a potential bug is detected; the program needs to be refined to a better design. Meanwhile, there will update and add more data to the database to increase the database capacity. Depending on the new keywords, responses, relevant data found that could be applied in this program; the program will always be improved and can handle more and more cases.

Future Scope

We plan to Integrate David with mobile using react native, to provide a synchronized experience between the two connected devices. Further, in the long run, David is planned to integrate with online server to make it more lightweight application for all the platforms and make it cross platform application. David is planned to be integrated with AI and ML for integrating the training mode.

This Intelligent Voice Assistant has an enormous and limitless scope in the future. Like Siri, Google Now and Cortana most popular personal voice assistants. The project will easily able to integrate with devices near future for a Connected Home using Internet of Things, voice command system and computer vision.

Voice assistants may be useful for library promotion and management as well. There are already tools available that allow libraries to create skills for voice assistants that list events at the library in local community calendars. Coding additional features that would allow patrons to hear hours and announcements, renew their items, hear what new items are available, or schedule a consultation with a librarian would be relatively simple. More complex tasks, such as searching databases and requesting interlibrary loans, are probably out of the scope of a tool that can only provide feedback via voice.

The user experience will become much better, making interactions richer and more natural. Their ability to process information on-device will open a whole new range of opportunities. All in all, virtual assistants will become more complex ecosystems that can support you in multiple areas of your everyday lives.

Conclusion

The personal voice assistant system presented in this project is very fundamental system with few features however the additional and advance feature may be introduced as future work of this project, in this report the design and implementation of an Intelligent Personal Voice Assistance is described. The project is built using available open source software modules with pycharm code community backing which can accommodate any updates in future. Speech recognition technology is a key technology which will provide a new way of human interaction with machine or tools.

The modular approach used in this project makes it more flexible and easy to integrate additional modules and features without disturbing the current system functionaries. It not only works on human commands but also it is designed for give responses to the user on the basis of query being asked or the words spoken by the user such as opening tasks and operations. This Intelligent Voice Assistant has an enormous and limitless scope in the future.

However, there are also several problems with the currently available voice assistant products. Privacy and security controls will need to be improved before voice assistants should be used for anything that requires confidentiality.

BIBLIOGRAPHY & REFERENCE

Websites referred

6. www.stackoverflow.com
7. www.pythonprogramming.net
8. www.codecademy.com
9. www.tutorialspoint.com
10. www.google.co.in
11. www.geeksforgeeks.org

YouTube Channels referred

- FreeCodeCamp
- Edureka!
- Codemy.com