VIRTUAL ASSISTANT

A PROJECT REPORT

Submitted by

MATHIVANAN M **KISHORE B**

513418104020 **513418104016**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING IN

COMPUTER SCIENCE & ENGINEERING

UNIVERSITY COLLEGEOF ENGINEERING KANCHEEPURAM

(A Constituent College of Anna University, Chennai)

ANNA UNIVERSITY : CHENNAI – 600 025

APRIL 2021

BONAFIDE CERTIFICATE

Certified that this project report titled "**VIRTUAL ASSISTANT**" is the bonafide work of "**MATHIVANAN M(513418104020) and KISHORE B(513418104016)**" of Computer Science & Engineering whose carried out this project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. V. Kavitha, M.E., Ph.D	**Mr.M.SURESH, M.E., HEAD OF**
DEPARTMENT,	**SUPERVISOR,**
Professor,	Teaching Fellow,
Department of CSE,	Department of CSE,
University College of Engineering	University College of Engineering
Kancheepuram – 631 552.	Kancheepuram – 631 552.

Submitted for the project viva-voice examination held on …………………

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are always thankful to our college Dean **Dr.P.SAKTHIVEL, M.E, Ph.D., PROFESSOR,** who endorsed us throughout this project.

Our heartfelt thanks to HOD **Dr.V.KAVITHA, M.E, Ph.D., PROFESSOR,** Department of Computer Science and Engineering, University College of Engineering, Kancheepuram, for the prompt and limitless help in providing the excellent infrastructure to do the project and to prepare the thesis.

We express our deep sense of gratitude to our guide **Mr.M.SURESH, M.E., Teaching Fellow, Department of Computer Science & Engineering,** for his invaluable support and guidance and encouragement for successful completion of this project. His vision and spirit will always inspire and enlighten us.

We express our sincere thanks to the project committee members **Dr.K.SELVABHUVANESWARI, M.E, Ph.D., ASSISTANT PROFESSOR, Mrs.T.KALA, M.E, ASSISTANT PROFESSOR, Mr.M.SURESH,M.E,**
TEACHING FELLOW ,Department of Computer Science and Engineering, Kancheepuram, for their invaluable guidance and technical Support.

We thank all the faculty members of Department of computer Science and Engineering for their valuable help and guidance. We are grateful to our family and friends for their constant support and encouragement. We thank the almighty whose showers of blessings made this project a reality.

ABSTRACT

The project aims to develop a personal virtual assistant for windows based system. Jarvis draws its inspiration from virtual assistants like Cortana for Windows, and Siri for iOS. It has been designed to provide a user-friendly interface for carrying out a variety of tasks by employing certain well-defined commands. Users can interact with the assistant either through voice commands or using keyboard input. As a personal assistant, Jarvis assists the end-user with day-to-day activities like general human conversation, searching queries in google or yahoo, searching for videos, playing songs, live weather conditions, word meanings, searching for medicine details and reminding the user about the scheduled events and tasks. The virtual assistant takes the voice input through our microphone and it converts our voice into computer understandable language and gives the required solutions and answers which are asked by the user. This assistant connects with the world wide web to provide results that the user has questioned. This project works on voice input and gives output through voice and displays the text on the screen. The main agenda of our virtual assistant is that it makes people smart and give instant and computed results.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

ACRONYM	ABBREVATIONS
UML	UNIFIED MODELING LANGUAGE
UI	USER INTERFACE
NLP	NATURAL LANGUAGE PROCESSING
API	APPLICATION PROGRAMMING INTERFACE
OS	OPERATING SYSTEM
GUI	GRAPHICAL USER INTERFACE
AI	ARTIFICIAL INTELLIGENCE
IoT	INTERNET OF THINGS
SAPI	SPEECH APPLICATION PROGRAMMING INTERFACE

CHAPTER 1
INTRODUCTION

1.1 OBJECTIVE OF THE PROJECT

The development of technology allows introducing more advanced solutions in everyday life. This makes work less exhausting for employees, and also increases the work safety. As the technology is developing day by day people are becoming more dependent on it, one of the mostly used platform is computer. We all want to make the use of these computers more comfortable, traditional way to give a command to the computer is through keyboard but a more convenient way is to input the command through voice. Giving input through voice is not only beneficial for the normal people but also for those who are visually impaired who are not able to give the input by using a keyboard. For this purpose, there is a need of a virtual assistant which can not only take command through voice but also execute the desired instructions and give output either in the form of voice or any other means.

A Virtual Assistant is the software that can perform task and provide different services to the individual as per the individual's dictated commands. This is done through a synchronous process involving recognition of speech patterns and then, responding via synthetic speech. Through these assistants a user can automate tasks ranging from but not limited to mailing, tasks management and media playback. It understands natural language voice commands and complete the tasks for the user. It is typically a cloud-based program that requires internet connected devices and/or applications to work. The technologies that power virtual assistants are machine learning, natural language processing and speech recognition platforms. It uses sophisticated algorithms to learn from data input and become better at predicting the end user's needs.

1.2 MOTIVATION

The main purpose of this project is to build a program that will be able to service to humans like a personal assistant. This is an interesting concept and many people around the globe are working it. Today, time and security are the two main things to which people are more sensitive, no one has the time to spoil; nobody would like their security breach, and this project is mainly for those kinds of people.

This system is designed to be used efficiently on desktops. Virtual Assistants software improves user productivity by managing routine tasks of the user and by providing information from an online source to the user. This project was started on the premise that there is a 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.

1.3 SCOPE OF THE PROJECT

Virtual Assistants will continue to offer more individualized experiences as they get better at differentiating between voices. However, it's not just developers that need to address the complexity of developing for voice as brands also need to understand the capabilities of each device and integration and if it makes sense for their specific brand. They will also need to focus on maintaining a user experience that is consistent within the coming years as complexity becomes more of a concern. This is because the visual interface with virtual assistants is missing. Users simply cannot see or touch a voice interface. Virtual Assistants are software programs that help you ease your day-to-day tasks, such as showing weather report, playing music etc. They can take commands via text (online chat bots) or by voice.

1.4 APPLICABILITY

The mass adoption of artificial intelligence in users' everyday lives is also fuelling the shift towards voice. The number of IoT devices such as smart thermostats and speakers are giving voice assistants more utility in a connected user's life. Smart speakers are the number one way we are seeing voice being used.

Many industry experts even predict that nearly every application will integrate voice technology in some way in the next 5 years. The use of virtual assistants can also enhance the system of IoT (Internet of Things). Twenty years from now, Microsoft and its competitors will be offering personal digital assistants that will offer the services of a full-time employee usually reserved for the rich and famous.

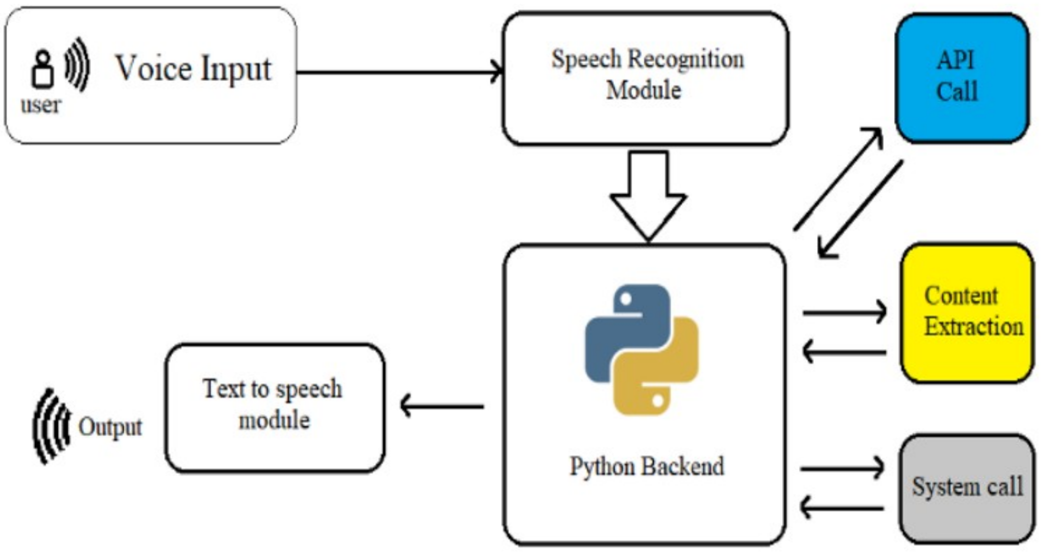

FIGURE 1.1 VIRTUAL ASSISTANT

CHAPTER 2

PRELIMINARIES

2.1 EXISTING SYSTEM

This project describes one of the most efficient ways for voice recognition. It overcomes many of the drawbacks in the existing solutions to make the Virtual Assistant more efficient. It uses natural language processing to carry out the specified tasks. It has various functionalities like network connection and managing activities by just voice commands. It reduces the utilization of input devices like keyboard.

This project describes the method to implement a virtual assistant for desktop using the APIs. In this module, the voice commands are converted to text through Google Speech API. Text input is just stored in the database for further process. It is recognized and matched with the commands available in database. Once the command is found, its respective task is executed as voice, text or through user interface as output.

2.1.1 DISADVANTAGES

- They propose a new detection scheme that gets two similar results which could cause confusions to the user on deciding the actual/desired output.
- Though the efficiency is high of the proposed module, the time consumption for each task to complete is higher and also the complexity of the algorithms would make it very tough to tweak it if needed in the future.

2.2 PROPOSED SYSTEM

1.QUERIES FROM THE WEB:

Making queries is an essential part of one’s life. We have addressed the essential part of a netizen’s life by enabling our voice assistant to search the web. Virtual Assistant supports a plethora of search engine like Google displays the result by scraping the searched queries.

2. ACCESSING NEWS:

Being up-to-date in this modern world is very much important. In that way news plays a big crucial role in keeping ourselves updated. News keeps you informed and also helps in spreading knowledge.

3. TO SEARCH SOMETHING ON WIKIPEDIA:

Wikipedia's purpose is to benefit readers by acting as a widely accessible and free encyclopaedia; a comprehensive written compendium that contains information on all branches of knowledge.

4. ACCESSING MUSIC PLAYLIST:

Music have remained as a main source of entertainment, one of the most prioritized tasks of virtual assistants. you can play any song of your choice. However, you can also play a random song with the help of a random module. Every time you command to play music, the Virtual Assistant will play any random song from the song directory.

5. OPENING CODE EDITOR:

Virtual Assistant is capable of opening your code editor or IDE with a single voice command.

2.2.1 ADVANTAGES

- Platform independence
- Increased flexibility
- Saves time by automating repetitive tasks
- Accessibility options for Mobility and the visually impaired
- Reducing our dependence on screens
- Adding personality to our daily lives
- More human touch
- Coordination of IoT devices
- Accessible and inclusive
- Aids hands free operation

CHAPTER 3

LITERATURE SURVEY

3.1 “VOICE ASSISTANT USING PYTHON”

YEAR: 2020

AUTHORS: Subhash Mani Kaushal, Megha Mishra

CONCEPT: Natural Language Processing.

3.2 “DESKTOP VOICE ASSISTANT”

YEAR: 2020

AUTHORS: Gaurav Agarwal, Harsha Gupta, Chinmay Jain

CONCEPT: Prerequisite APIs For Virtual Assistant.

3.3 “SMART PYTHON CODING THROUGH VOICE RECOGNITION”

YEAR: 2019

AUTHORS: M. A. Jawale, A. B. Pawar, D. N. Kyatanavar

CONCEPT: User experience field for better programming Integrated Development Environment Development (IDE).

3.4 “VPA: VIRTUAL PERSONAL ASSISTANT”

YEAR: 2018

AUTHORS: Nikita Saibewar, Yash Shah, Monika Das

CONCEPT: Implementation of Functionalities of VPA.

3.5 “SURVEY ON VIRTUAL ASSISTANT”

YEAR: 2018

AUTHORS: Amrita Sunil Tulshan, Sudhir Namdeorao

CONCEPT: Functionalities of Existing IVAs.

3.6 “PERSONAL ASSISTANT WITH VOICE RECOGNITION INTELLIGENCE”

YEAR: 2017

AUTHORS: Dr. Kshama,V Kulhalli, Dr. Kotrappa Sirbi, Mr. Abhijit J. Patankar

CONCEPT: Developing a Personal Assistant which has capability to work with and without Internet Connectivity.

CHAPTER 4

SYSTEM SPECIFICATIONS

4.1 HARDWARE REQUIREMENTS

- Processor - Intel Pentium 4
- RAM - 512 MB
- Hardware capacity:80GB
- Monitor type- 15inch colour monitor
- CD-Drive type- 52xmax
- Mouse
- Microphone
- Personal Computer / Laptop

4.2 SOFTWARE REQUIREMENTS

- Operating System - Windows
- Simulation Tools - Visual Studio Code
- Python - Version 3.9.6
- Packages -
    1. Pyttsx3
    2. Speech Recognition
    3. Wikipedia
    4. Pyaudio
    5. Webbrowser

CHAPTER 5

SYSTEM DESIGN

5.1 ARCHITECTURE DIAGRAM

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. After going through the above process, we have successfully enabled the model to understand the features.

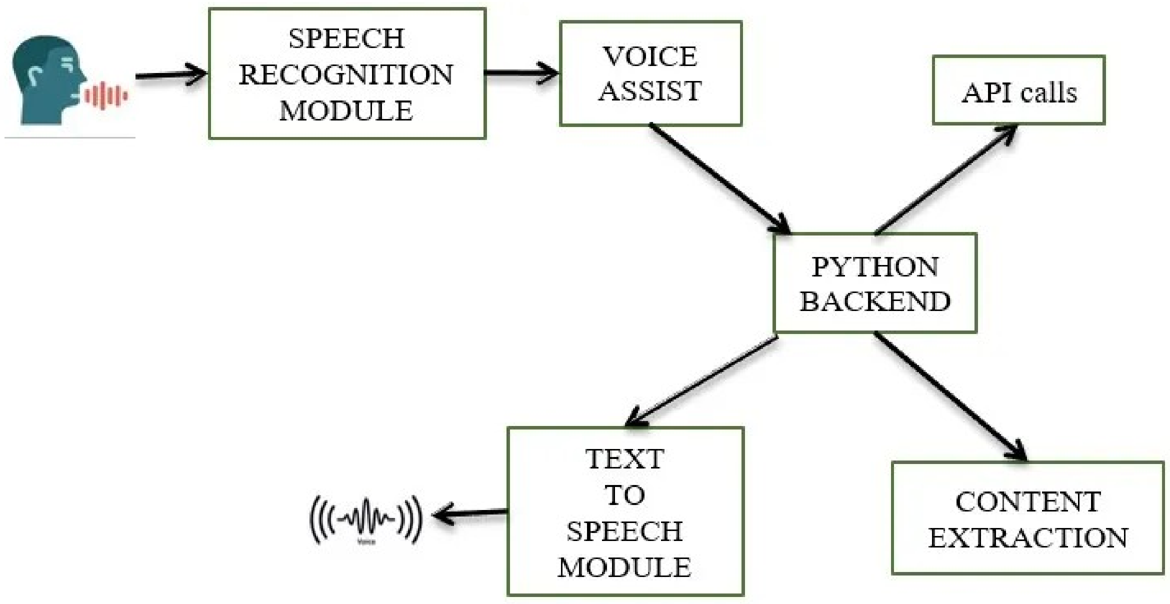

FIGURE 5.1 SYSTEM ARCHITECTURE DIAGRAM

5.2 UML DIAGRAM

The Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

UML defines several models for representing systems

1. The class model captures the static structure
2. The state model expresses the dynamic behavior of objects
3. The use case model describes the requirements the requirements of the user
4. The interaction model represents the scenarios and messages flows
5. The implementation model shows the work unit

ADVANTAGES

- Most used and flexible
- Development time is reduced
- Provides standard for software development
- It has large visual elements to construct and easy to follow

5.2.1 USE CASE DIAGRAM

In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. In this project there is only one user. The user queries command to the system. System then interprets it and fetches answer. The response is sent back to the user.

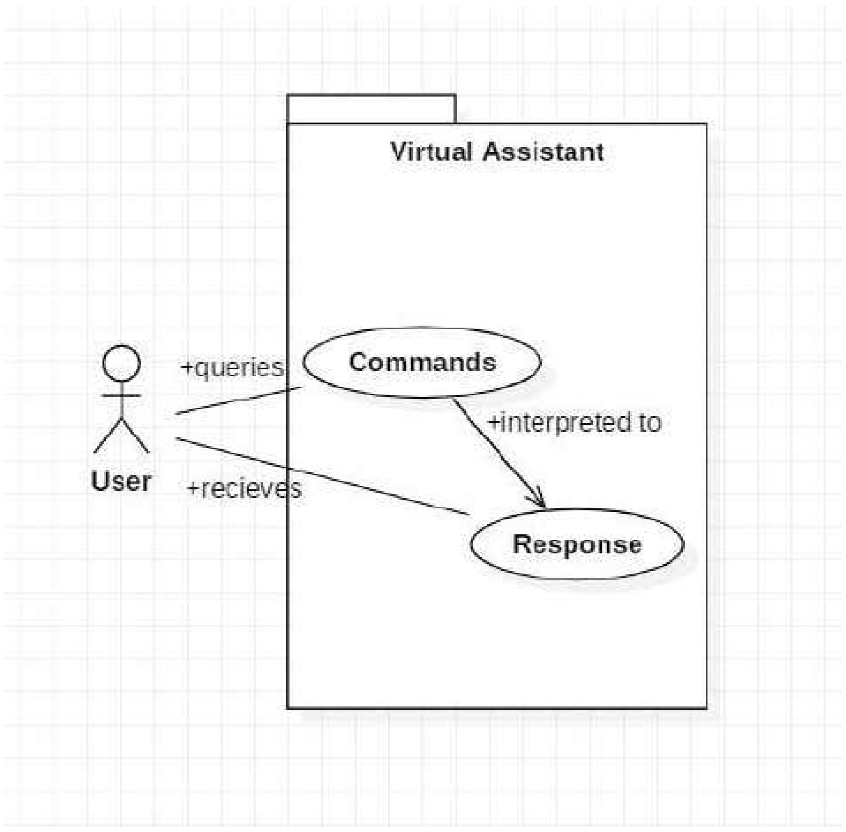

FIGURE 5.2 USE CASE DIAGRAM

5.2.2 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class user has 2 attributes command that it sends in audio and the response it receives which is also audio. It performs function to listen the user command. Interpret it and then reply or sends back response accordingly. Question class has the command in string form as it is interpreted by interpret class. It sends it to general or about or search function based on its identification. The task class also has interpreted command in string format.

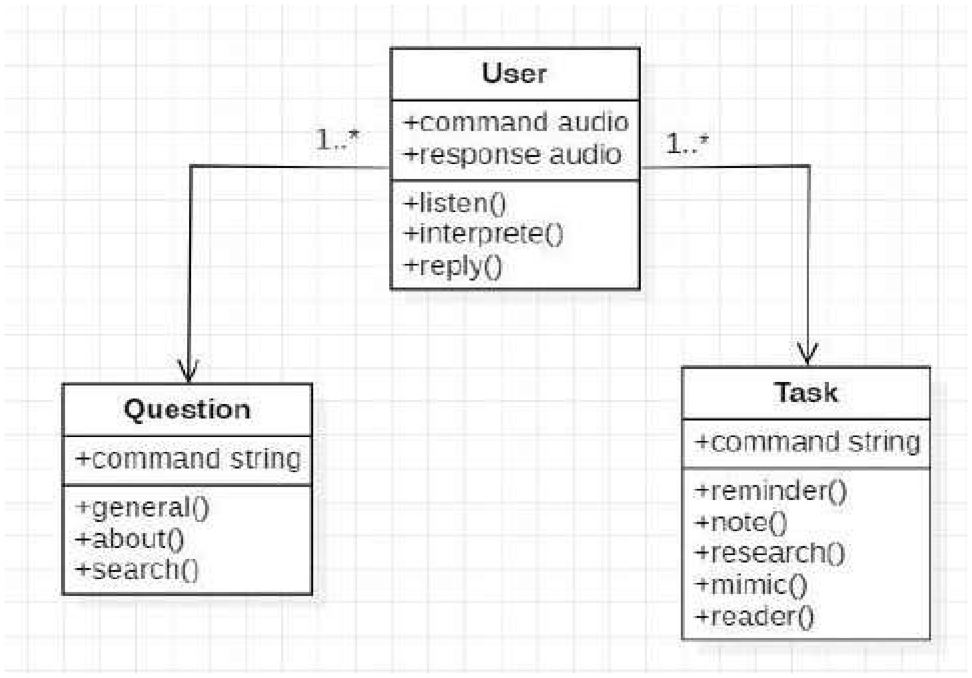

FIGURE 5.3 CLASS DIAGRAM

5.2.3 ACTIVITY DIAGRAM

An activity diagram is a behavioral diagram. It depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

Initially, the system is in idle mode. As it receives any wakeup call it begins execution. The received command is identified whether it is a question or task to be performed. Specific action is taken accordingly. After the question is being answered or the task is being performed, the system waits for another command. This loop continues unless it receives a quit command.

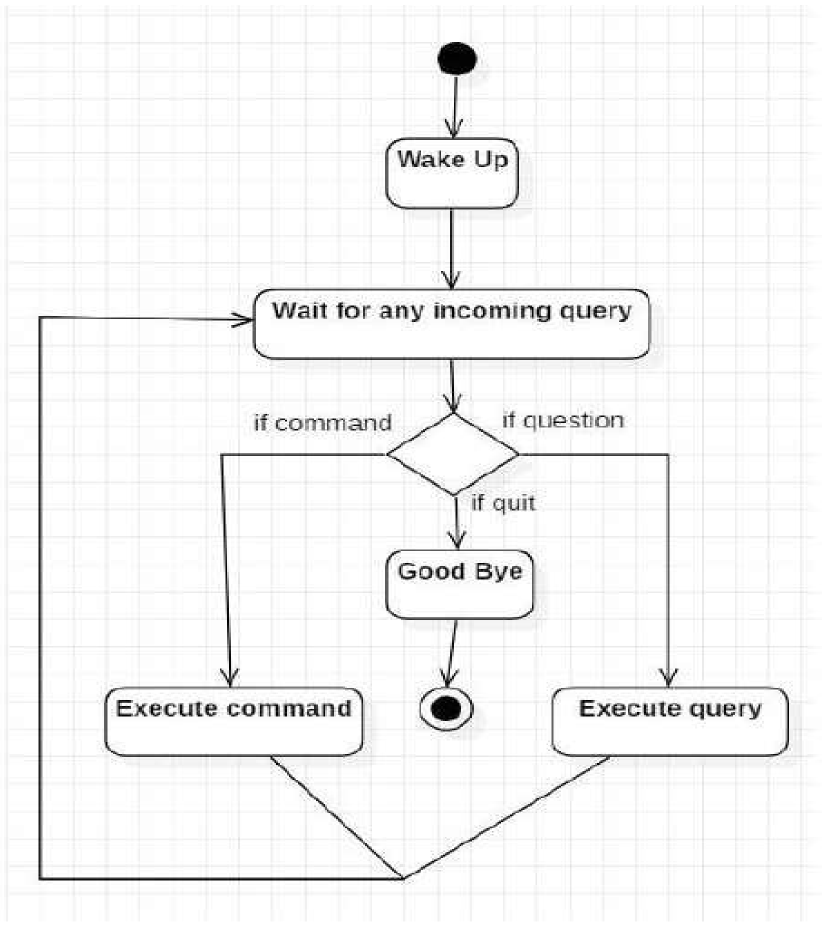

FIGURE 5.4 ACTIVITY DIAGRAM

5.2.4 SEQUENCE DIAGRAM

A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction.

The below 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.

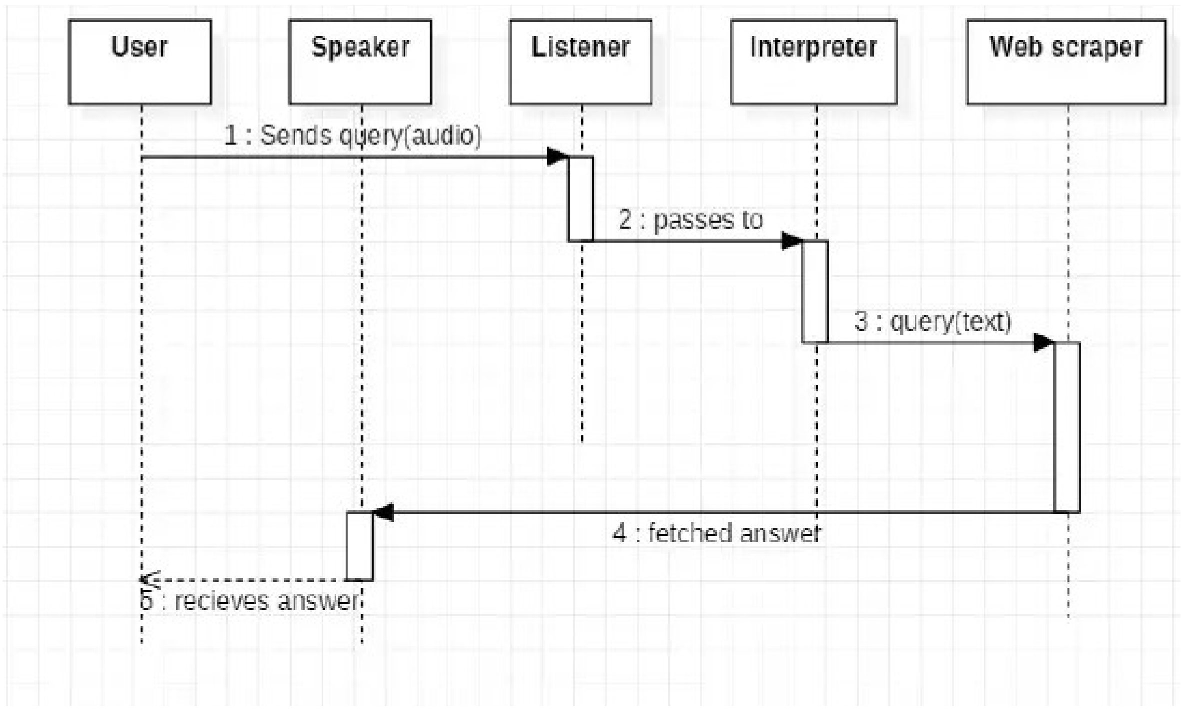

FIGURE 5.5(a) SEQUENCE DIAGRAM FOR QUERY- RESPONSE

The user sends command to virtual assistant in audio form. The command is passed to the interpreter. It identifies what the user has asked and directs it to task executer. If the task is missing some info, the virtual assistant asks user back about it. The received information is sent back to task and it is accomplished. After execution feedback is sent back to user.

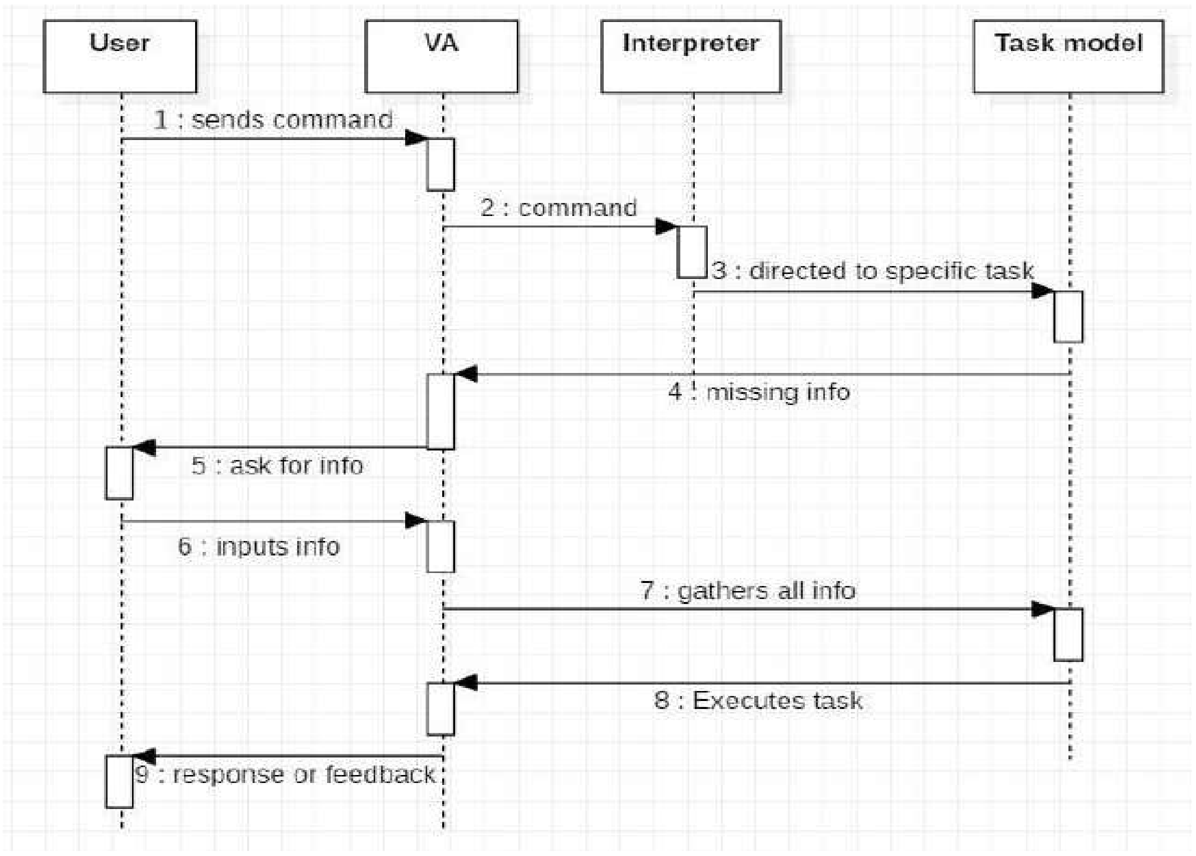

FIGURE 5.6(b) SEQUENCE DIAGRAM FOR TASK EXECUTION

CHAPTER 6
SYSTEM IMPLEMENTATION

6.1 MODULES

- Pyttsx3
- Sapi5
- Speech recognition
- Pyaudio
- Wikipedia
- Webbrowser

6.2 MODULE DESCRIPTION

6.2.1 Pyttsx3 (Python Text to Speech)

A python library that will help us to convert text to speech. It is a cross-platform Python wrapper for text-to-speech synthesis. It is a Python package supporting common text-to-speech engines on MacOS X, Windows, and Linux. It works for both Python2.x and 3.x versions. Its main advantage is that it works offline

6.2.2 Sapi5 (Speech Application Programming Interface)

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. Many versions (although not all) of the speech recognition and synthesis engines are also freely redistributable. SAPI 5 however was a completely new interface, released in 2000. Since, then several sub-versions of this API have been released.

6.2.3 Speech recognition

Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text.

Speech Recognition is an important feature in several applications used such as home automation, artificial intelligence, etc.

Recognizing speech needs audio input, and Speech Recognition makes it really simple to retrieve this input. This is a library for performing speech recognition, with support for several engines and APIs, online and offline.

6.2.4 Pyaudio

To access your microphone with Speech Recognizer, you'll have to install the **PyAudio** package. PyAudio provides Python bindings for Port Audio, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a varity of platforms.

6.2.5 Wikipedia

Wikipedia is a Python library that makes it easy to access and parse data from Wikipedia. It gets article summaries, get data like links and images from a page, and more. This module provides developers code-level access to the entire Wikipedia reference.

6.2.6 Webbrowser

The webbrowser module provides a high-level interface to allow displaying Web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.

CHAPTER 7

SYSTEM STUDY

7.1 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 of the proposed system.

Three key considerations involved in the feasibility analysis are:

- Economical feasibility
- Technical feasibility
- Social feasibility

7.1.1 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. User also would have to pay for microphones and speakers. Again, they are cheap and available.

7.1.2 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 what system speaks. These are very cheap now a days and everyone generally possess them.

Besides, system needs internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.

7.1.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 8

SYSTEM TESTING

8.1 TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub – assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

8.2 TYPES OF TESTING UNIT TESTING

8.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

8.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical 29 requirements, system documentation, and user manuals. Functional testing is centred on the following items:

- **Valid input**: identified classes of valid input must be accepted identified classes of valid input must be accepted.
- **Invalid output**: identified classes of valid input must be accepted.
- **Functions:** Identified functions must be exercised
- **Output**: identified classes of application outputs must be exercised.

8.2.4 SYSTEM TESTING

System testing ensures that the entire integrated software system. It tests a configuration to ensure known and predictable results. System testing is based on the process descriptions, flows, emphasizing pre-driven process links and integration points.

8.2.5 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the 30 software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

8.2.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document.

8. 3 TEST RESULTS

All the test cases mentioned above have passed successfully. No defects encountered.

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 CONCLUSION

Through this virtual assistant, we have automated various services using a single line command. It eases most of the tasks of the user like searching the web, playing music and doing Wikipedia searches. We aim to make this project a complete server assistant and make it smart enough to act as a replacement for a general server administration. The project is built using available open-source software modules with visual studio code community backing which can accommodate any updates in future. The modular nature of this project makes it more flexible and easier to add additional features without disturbing current system functionalities. It not only works on human commands but also give responses to the user based on the query being asked or the words spoken by the user such as opening tasks and operations. The application should also eliminate any kind of unnecessary manual work required in the user life of performing every task.

9.2 FUTURE WORK

The virtual assistants which are currently available are fast and responsive but we still have to go a long way. The understanding and reliability of the current systems need to be improved a lot. The assistants available nowadays are still not reliable in critical scenarios. The future plans include integrating our virtual assistant with mobile using React Native to provide a synchronised experience between the two connected devices. Further, in the long run, our virtual assistant is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general Server Administrator does. The future of these assistants will have the virtual assistants incorporated with Artificial Intelligence which includes Machine Learning, Neural Networks, etc. and IoT.

APPENDIX – 1

SOURCE CODE

```
import pyttsx3 #pip install pyttsx3
import speech_recognition as sr #pip install speechRecognition
import datetime
```

```python
import wikipedia #pip install wikipedia
import webbrowser
import os
import sys
import time
import smtplib

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
# print(voices[1].id)
engine.setProperty('voice', voices[0].id)


def speak(audio):
    engine.say(audio)
    engine.runAndWait()


def wishMe():
    hour = int(datetime.datetime.now().hour)
    if hour>=0 and hour<12:
        speak("Good Morning!")

    elif hour>=12 and hour<18:
        speak("Good Afternoon!")

    else:
        speak("Good Evening!")

    speak("I am Jarvis Sir. Please tell me how may I help you")

def takeCommand():
    #It takes microphone input from the user and returns string output

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold = 1
        audio = r.listen(source)

    try:
        print("Recognizing...")
        query = r.recognize_google(audio, language='en-in')
        print(f"User said: {query}\n")

    except Exception as e:
        # print(e)
        print("Say that again please...")
        return "None"
    return query
```

```
if __name__ == "__main__":
    wishMe()
    while True:
    # if 1:
        query = takeCommand().lower()

        # Logic for executing tasks based on query
        if 'wikipedia' in query:
            speak('Searching Wikipedia...')
            query = query.replace("wikipedia", "")
            results = wikipedia.summary(query, sentences=2)
            speak("According to Wikipedia")
            print(results)
            speak(results)


        elif 'open youtube' in query:
            webbrowser.open("youtube.com")

        elif 'open google' in query:
            webbrowser.open("google.com")

        elif 'open stackoverflow' in query:
            webbrowser.open("stackoverflow.com")

        elif 'news' in query:
            webbrowser.open("bbc.com")
            print("Here some headlines from the newspaper from BBC news, Happy
reading!")
            speak("Here some headlines from the newspaper from BBC news, Happy
reading!")

        elif 'play music' in query:
            music_dir = "C:/Users/dgsou/Music/fav-music"
            songs = os.listdir(music_dir)
            print(songs)
            os.startfile(os.path.join(music_dir, songs[0]))

        elif 'the time' in query:
            strTime = datetime.datetime.now().strftime("%H:%M:%S")
            speak(f"Sir, the time is {strTime}")

        elif 'open code' in query:
            codePath = "C:/Users/dgsou/AppData/Local/Programs/Microsoft VS Cod
e/Code.exe"
            os.startfile(codePath)

        elif 'goodbye' in query:
            speak("Goodbye! Have a nice day")
            print("Goodbye! Have a nice day")
            time.sleep(2)
            sys.exit()
```

APPENDIX – II

SNAPSHOTS

CHAPTER 10

REFERENCES

1. Abhay Dekate, Chaitanya Kulkarni, Rohan Killedar, “Study of Voice Controlled Personal Assistant Device”, International Journal of Computer Trends and Technology (IJCTT) – Volume 42 Number 1 – December 2016.

2. Deny Nancy, Sumithra Praveen, Anushria Sai, M.Ganga, R.S.Abisree, “Voice Assistant Application for a college Website”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7,April 2019.

3. Deepak Shende, Ria Umahiya, Monika Raghorte, Aishwarya Bhisikar, Anup Bhange, “AI Based Voice Assistant Using Python”, Journal of Emerging Technologies and Innovative Research (JETIR), February 2019, Volume 6.

4. Dr.Kshama V.Kulhalli, Dr.Kotrappa Sirbi, Mr.Abhijit J. Patankar, “Personal Assistant with Voice Recognition Intelligence”, International Journal of Engineering Research and Technology. ISSN 0974- 3154 Volume 10, Number 1 (2017).

5. Isha S. Dubey, Jyotsna S. Verma, Ms.Arundhati Mehendale, “An Assistive System for Visually Impaired using Raspberry Pi”, International Journal of Engineering Research & Technology (IJERT), Volume 8, May-2019.

6. Kishore Kumar R, Ms. J. Jayalakshmi, Karthik Prasanna, “A Python based Virtual Assistant using Raspberry Pi for Home Automation”, International Journal of Electronics and Communication Engineering (IJECE), Volume 5, July 2018.

7. M. A. Jawale, A. B. Pawar, D. N. Kyatanavar, “Smart Python Coding through Voice Recognition”, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8, August 2019.

8. Rutuja V. Kukade, Ruchita G. Fengse, Kiran D. Rodge, Siddhi P. Ransing, Vina M. Lomte, “Virtual Personal Assistant for the Blind”, International

Journal of Computer Science and Technology (JCST), Volume 9, October - December 2018.

9. Tushar Gharge, Chintan Chitroda, Nishit Bhagat, Kathapriya Giri, "AI-Smart Assistant", International Research Journal of Engineering and Technology (IRJET), Volume: 06, January 2019.

10. Veton Kepuska, "Next-Generation of Virtual Personal Assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google 11. Home)", PyCon, Cleveland, 2018