

## ABSTRACT

---

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 through voice commands. As we know Python is an emerging language so it becomes easy to write a script for Desktop Assistant in Python. The instructions for the assistant can be handled as per the requirement of user. SpeechRecognition is the process of converting speech into text. This is commonly used in desktop assistant like Alexa, Siri, etc. In Python there is an API called SpeechRecognition which allows to convert speech into text. It was an interesting task to make my assistant. In the current scenario, advancement in technologies are such that they can perform any task with same effectiveness or can say more effectively than us. By making this project, I realized that the concept of AI in every field is decreasing human effort and saving time. Functionalities of this project include:

1. It can search on Wikipedia.
2. It can search and open on Youtube.
3. It can open websites such as Stackoverflow.
4. It can open Google.
5. It can open command prompt, notepad, etc.
6. It can play music.
7. It can give current time.
8. It can wish good morning, good afternoon, etc.
9. It can play movie online.
10. It can have some basic conversation.

## **Chapter 1**

# **INTRODUCTION**

---

### **1.1 Introduction**

Upcoming trending technologies such as virtual reality, augmented reality, voice interaction, IOT etc are changing the way people engage with the world and transforming digital experiences. Voice control is one of important development of human-machine interaction, which was possible because of advancement in Artificial Intelligence. In current era, we are able to train our machine to do their tasks by themselves or to think like humans using technologies like Artificial Intelligence, Machine Learning, Neural Networks, etc. we can talk to our machines with the help of virtual assistants. In recent time great appearance of voice assistants such as Apple's Siri, Google's Assistant, Microsoft's Cortana and Amazon's Alexa have been noticed due to heavy use of smartphones. Voice assistants uses technologies like voice recognition, speech synthesis, and Natural Language Processing (NLP) to provide various services which help users to perform their task using their machine by just giving commands in voice format and also with the help of Voice Assistant there will be no need to write the commands again and again for performing particular task. Virtual assistants are very useful for old generation people, people with disabilities or special cases, small children who don't know to operate machines or smart gadgets, by making them sure that their interaction with machine is not difficult anymore and also enable them to perform Multitasking.

Artificial Intelligence when used with machines, it shows us the capability of thinking like humans. In this, a computer system is designed in such a way that typically requires interaction from human. As we know Python is an emerging language so it becomes easy to write a script for Desktop Assistant in Python. The instructions for the assistant can be handled as per the requirement of user. Speech Recognition is the Alexa, Siri, etc. In Python there is an API called SpeechRecognition which allows us to convert speech into text. It was an interesting task to make my own assistant. As the voice assistant is using Artificial Intelligence hence the result that it is providing are highly accurate and efficient.

The assistant can help to reduce human effort and consumes time while performing any task, they removed the concept of typing completely and behave as another individual to whom we are talking and asking to perform task. The assistant is no less than a human assistant but we can say that this is more effective and efficient to perform any task. The libraries and packages used to make this assistant focuses on the time complexities and reduces time. The functionalities of this project include, It can search on Wikipedia, It can search and open on Youtube, It can open websites such as Stackoverflow, It can open Google, It can open command prompt, notepad, etc, It can play music, It can give current time, It can wish good morning, good afternoon, etc, It can play movie online, It can have some basic conversation. Tools and technologies used are PyCharm IDE for making this project, and I created all py files in PyCharm. Along with this I used following modules and libraries in my project. pyttsx3, SpeechRecognition, Datetime, Wikipedia, pywhatkit, pyjokes, etc. AI Desktop assistant, also known as a voice or virtual assistant, is a device that uses voice recognition technology, natural language processing, and Artificial Intelligence (AI) to respond to people.

Through technology, the device aggregates user messages, breaks them down, rates them, and gives meaningful feedback in return. Artificial intelligence can bring real conversations. Desktop assistants, understand natural language voice commands and performs tasks for users. Desktop assistants are typically cloud-based programs that require internet-connected devices and/or applications to function. The technologies that power Desktop assistants require vast amounts of knowledge, powering the platforms, as well as machine learning, language communication processes, and speech recognition arena. There are dedicated devices to provide Desktop assistance. The most stylish on the market from Amazon, Google and Microsoft having Alexa, Google Siri and Cortana as AI voice assistants respectively given by each company.

Some Reasons why there is necessity of Desktop assistants:

There are lots of reason why this verbal voice command application is in need in real time situations. Some of them are given below.

- To enable a highly engaging user experience: Voice assistance engages users like no other interface. Users can speak to the applications naturally to ask for whatever they'd like.
- To make application frustration free: We have to touch, type and mouse in the existing machine system to getting our work done, which are makes user frustrated sometimes. By using voice assistant users can directly ask what they wanted to get done.
- To personalize your app experience for every user: Voice assistants are actually able to respond for every user based on their locality, language and preferences.
- To Remove Language Barriers: Voice Assistant technology are blended with Translation services which helps users to handle them in their own language without concerning about language barriers which allows them to interact more freely with voice assistant.



Fig1.1. Voice assistant logo

We know some of the Desktop assistants, like Google's Google Assistant, Apple's Siri, Amazon's Alexa, and Microsoft's Cortana this method is specifically designed to work effectively on desktops. Personal Assistant application code ameliorates users' productivity by organizing routine tasks and by dispensing data from an internet supply to the user. This project started on the assumption that there is an adequate amount of openly obtainable information & data in the Internet that can be utilized to make a virtual assistant that has ability to building intelligent decisions for the regular user activities. In this modern world, almost everything is getting digitalized. We have smartphones with us and it is equal to having the world at your fingertip. Nowadays we are not even using our fingers to type or touch. We just speak about the task and task gets done by virtual assistant. The virtual assistant must also support some specialized tasks such as translation, weather reports, buzzing, searching, browsing, any task, etc. Voice searches have subordinates' text-based web searches conducted through smartphone devices that just overtaken those carried out by using a computer system and the analysts are foretelling that 50% of searches will be via speech\voice by 2020. Virtual assistants are getting out smarter than ever.

Let your Virtual assistant work for you, pick out information and deliver a good responses. The main aim of this project is to make a program that will be able to service humans like a personal assistant. This software aims at application-software is to perform and execute the user's tasks for certain commands, given in either speech or text. It will ease the work of the user.



Fig2.1 Microphone

## 1.2 Objective of Project

Main objective of building personal assistant software (a desktop 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 desktop assistant is to answer questions that users may have. This may be done in a business environment, for example, on the business website, with a chat interface. On the mobile platform, the intelligent voice assistant is available as a call-button operated service where a voice asks the user “What can I do for you?” and then responds to verbal input.

Desktop assistants can tremendously save you time. We spend hours in online research and then making the report in our terms of understanding. JARVIS can do that for you. Provide a topic for research and continue with your tasks while JARVIS 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 JARVIS in advance about your tests and she reminds you well in advance so you can prepare for the test.

One of the main advantages of voice searches is their rapidity. In fact, voice is reputed to be four times faster than a written search: whereas we can write about 40 words per minute, we are capable of speaking around 150 during the same period of time<sup>15</sup>. In this respect, the ability of personal assistants to accurately recognize spoken words is a prerequisite for them to be adopted by consumers.

Purpose of Desktop assistant is to being capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news. Desktop assistants enable users to speak natural language voice commands in order to operate the device and its apps. There is an increased overall awareness and a higher level of comfort demonstrated specifically by millennial consumers. In this ever-evolving digital world where speed, efficiency, and convenience are constantly being optimized, it's clear that we are moving towards less screen interaction.

## **1.2 Problem Definition**

We are all well aware about Cortana, Siri, Google Assistant and many other virtual assistants which are designed to aid the tasks of users in Windows, Android and iOS platforms. But to our surprise, there's no such complete virtual assistant available for Core Windows platform consisting of 70% of the users. So, this is actually a major problem for users where there could be internet instability, server problems and places where internet is not accessible

Usually, user needs to manually manage multiple sets of applications to complete one task. For example, a user trying to make a travel plan needs to check for airport codes for nearby airports and then check travel sites for tickets between combinations of airports to reach the destination. There is need of a system that can manage tasks effortlessly.

We already have multiple voice assistants. But we hardly use it. 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 voice assistant that can understand English in Indian accent and work on desktop system.

When a desktop 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 desktop assistant learning undesired bad behaviors.

They require large amount of information to be fed in order for it to work efficiently. Voice 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 there can be multiple solutions to paths, and the it should be able to consider user preferences, other active tasks, priorities in order to recommend a particular plan.

## Chapter 2

# LITERATURE SURVEY

---

### 2.1 Related Work

- ‘Voice Assistant using Python’  
Year: 2020  
Authors: Subhash Mani Kaushal, Megha Mishra  
Concept: Natural Language Processing
- ‘Desktop Voice Assistant’  
Year: 2020  
Authors: Gaurav Agarwal, Harsha Gupta, Chinmay Jain  
Concept: Prerequisite API’s for Virtual Assistant
- ‘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).
- ‘VPA: Virtual Personal Assistant’  
Year: 2018  
Authors: Nikita Saibewar, Yash Shah, Monika Das  
Concept: Implementation of Functionalities of VPA.
- ‘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.



## 2.2 Requirement Analysis

The complete analysis is followed below with the help of feasibility study.

### 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.

- 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 JIA, make sure you have a steady internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.
- 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.
- 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 microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, JIA won't cost too much.
- Organizational feasibility: This shows the management and organizational structure of the project. This project is not built by a team. The management tasks are all to be carried out by a single person. That won't create any management issues and will increase the feasibility of the project

### Software Requirements:

- JetBrains PyCharm Community Edition 2022.3.3
- Build#PC-191.6605.12, built on April 3, 2019
- JRE:11.0.2+9-b159.34amd64

### Hardware Requirements Operating System:

- Windows 10 Processor: intel core i5 Disk space:1Gb

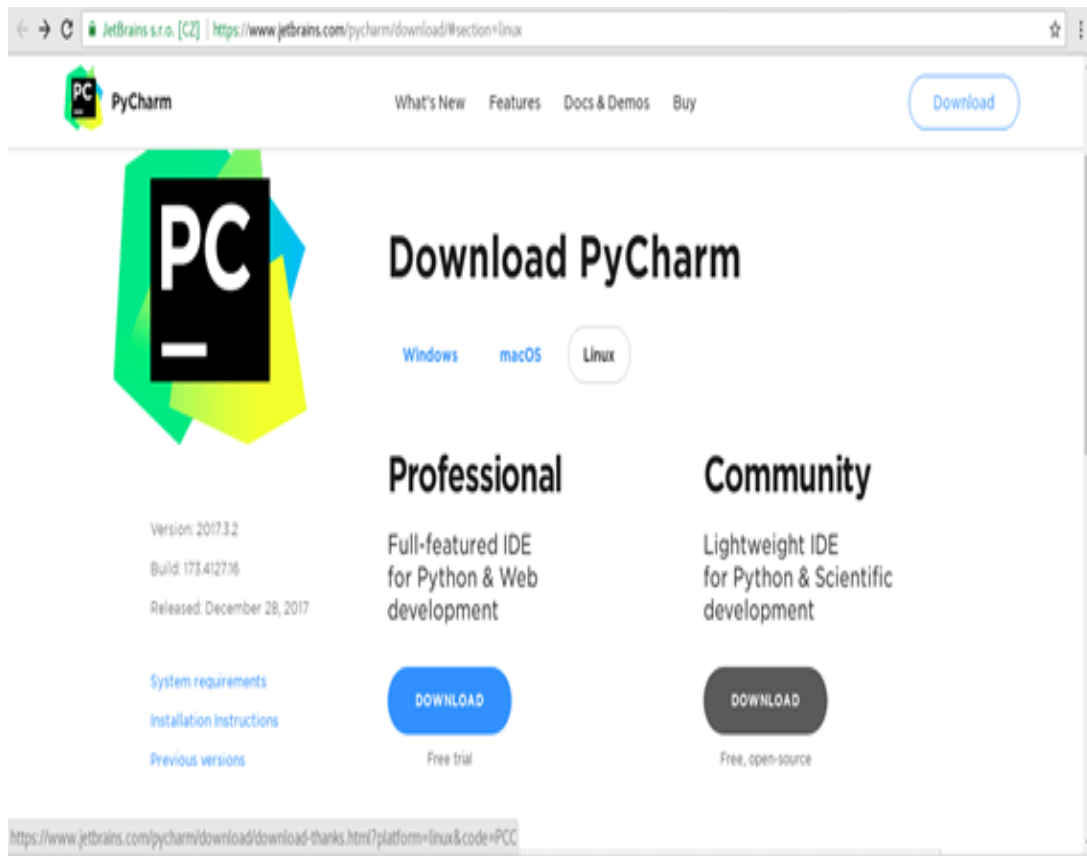


Fig3.1 Installation of PyCharm

## Chapter 3

### SYSTEM ANALYSIS & DESIGN

System Analysis is about complete understanding of existing systems and finding where the existing system fails. The solution is determined to resolve issues in the proposed system. It defines the system. The system is divided into smaller parts. Their functions and inter relation of these modules are studied in system analysis.

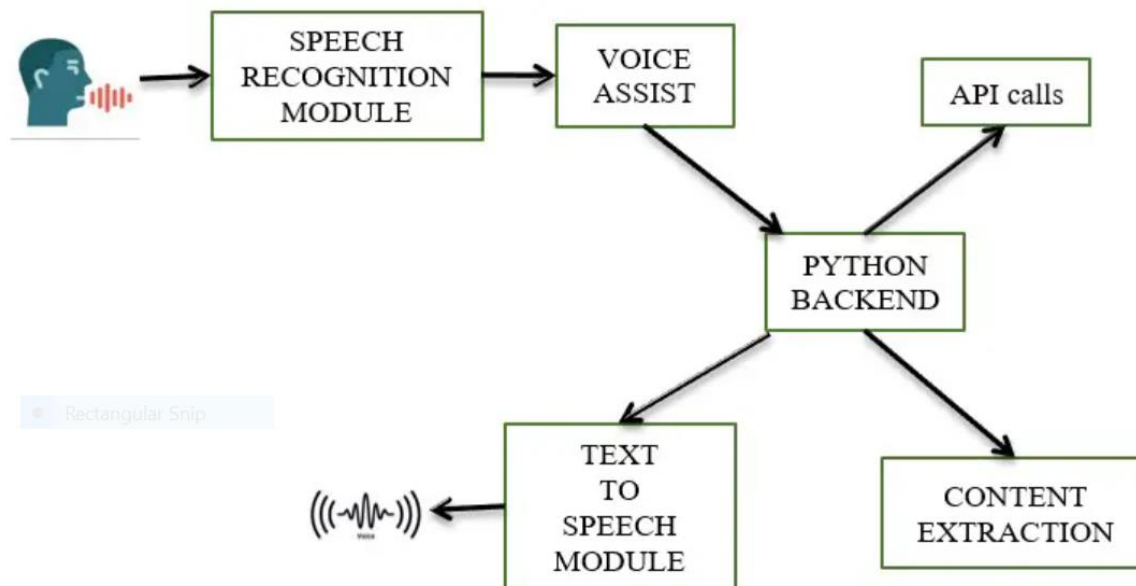


Fig4.1 Detail Workflow

#### 3.1 Type of SDLC Model Used:

There are a lot of proposed software/game development lifecycle models which follow a specific life cycle during the process of development. These models are also termed as Process & Development Models. All these models follow a chain of steps in a circular formation, which is unique to its type for ensuring success in its development stage.

The list of some of the famous development models that all corporate firms and developers follow are:

- WaterFall Model
- Spiral Model

- **Waterfall Model**

It is the oldest and most frequently used model and follows simple and straight-forward methodologies - according to which first complete one phase, then move on to the next phase (no going backwards). Every stage of waterfall model depends on the information passed on from the previous stage. It is easier to understand and to manage effectively. This model was very popular during early(1980's to mid 90's) developments when the requirements were constant through the development. But these days, requirements change every day, hence following this model is not a good choice. It can be used for small mini projects.

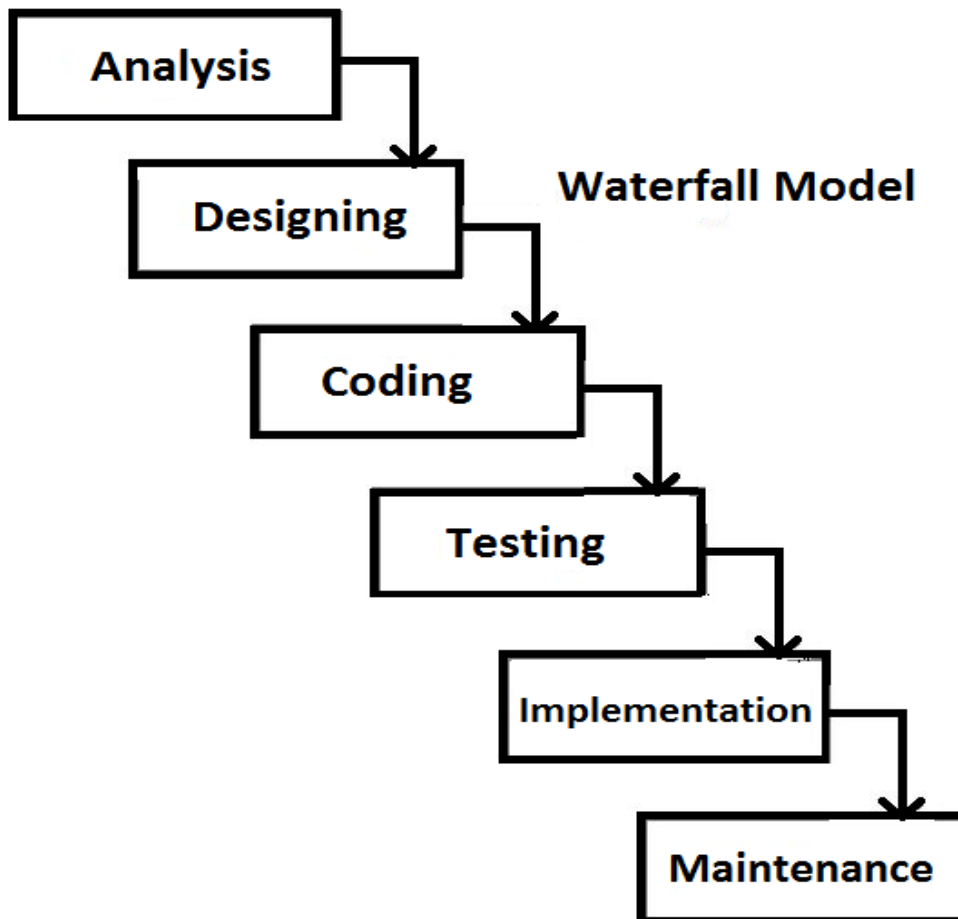


Fig5.1 Waterfall Model

- **Spiral Model**

This one is a flexible model. The spiral model has a repetitive approach, going forward in a circular manner where the project passes through four phases over and over in the form of a **spiral**, until it reaches the completion, hence allowing several rounds of refinement.

In development, the typical steps in a Spiral Lifecycle Model are:

1. Design and Planning
2. Implement the plan or in other words code the Game.
3. Play Test: This involves playing the game and analysing it for improvements, looking for bugs/issues etc.
4. Evaluation of the current progress. Understanding what we did right, what we did wrong, and with new point of observations, move back to step 1.

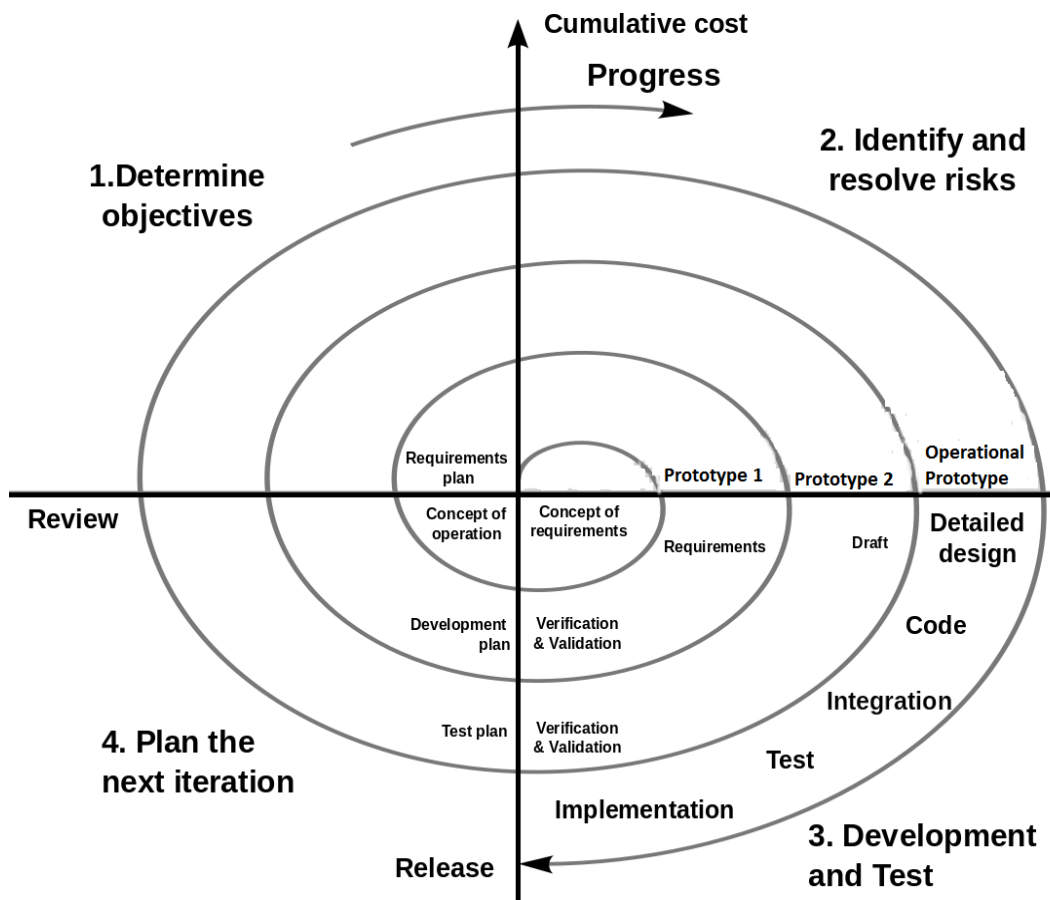


Fig6.1 Spiral Model

### **3.2 UML Diagrams**

The Unified Modeling Language is a general purpose, development, 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

- The class model captures the static structure.
- The state model expresses the dynamic behavior of objects.
- The use case model describes the requirements of the user.
- The interactions model represents the scenarios and messages flows.
- The implementation model shows the work unit.

#### 1. Use Case Diagram

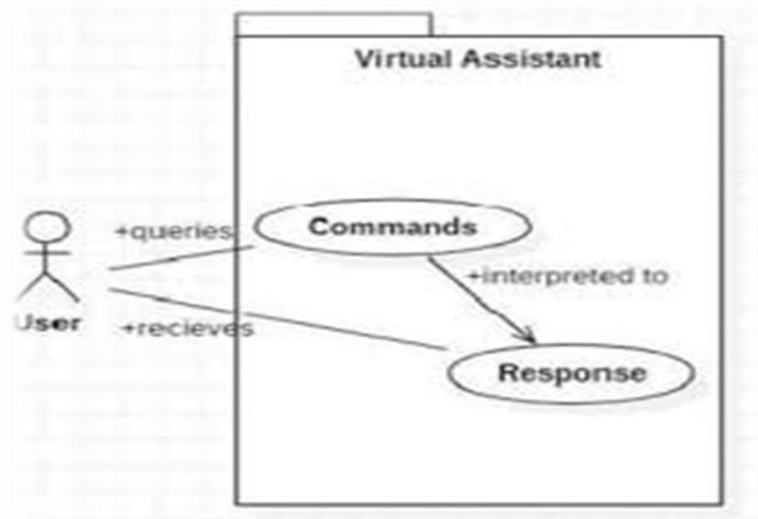


Fig7.1 Use Case Diagram

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.

## 2. Data Flow Diagram

### 2.1.DFD Level 0 (Context Level Diagram)

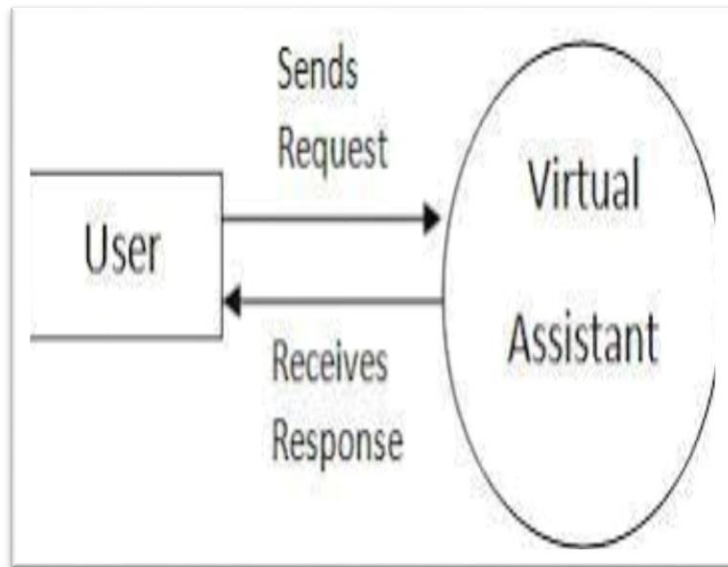


Fig8.1 DFD Level 0

### 2.2.DFD Level 1

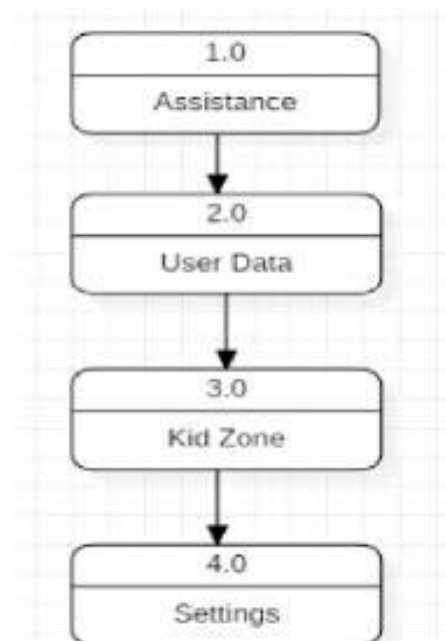


Fig8.2 DFD Level 1

### 3. Component Diagram

The main component here is the Virtual Assistant. It provides two specific service, executing Task or Answering your question.

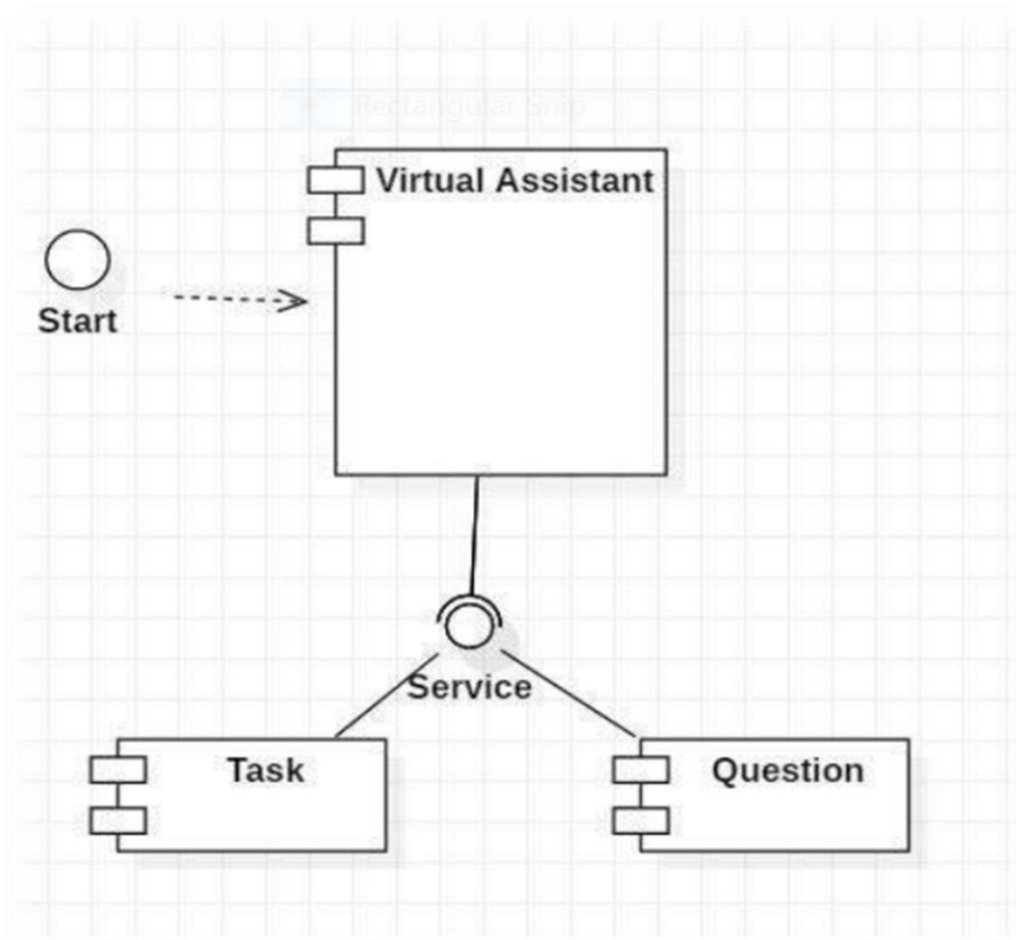


Fig9.1 Component Diagram



#### 4. Activity Diagram

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 questionnaire or a 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 quit command. At that moment, it goes back to sleep.

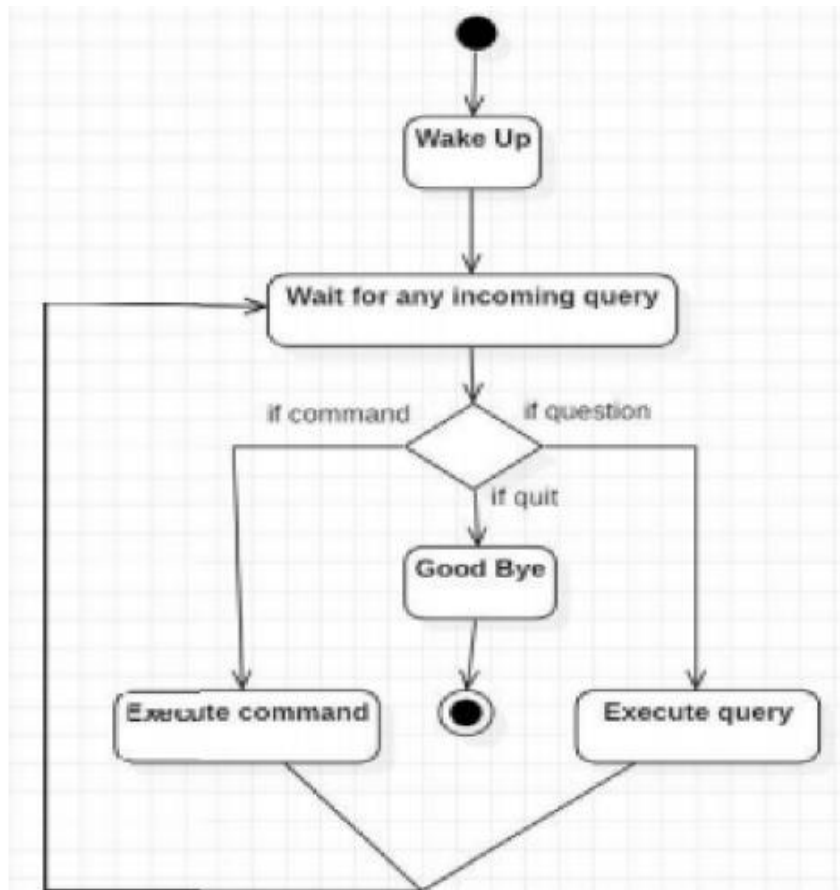


Fig10.1 Activity Diagram

## 5. Sequence Diagram

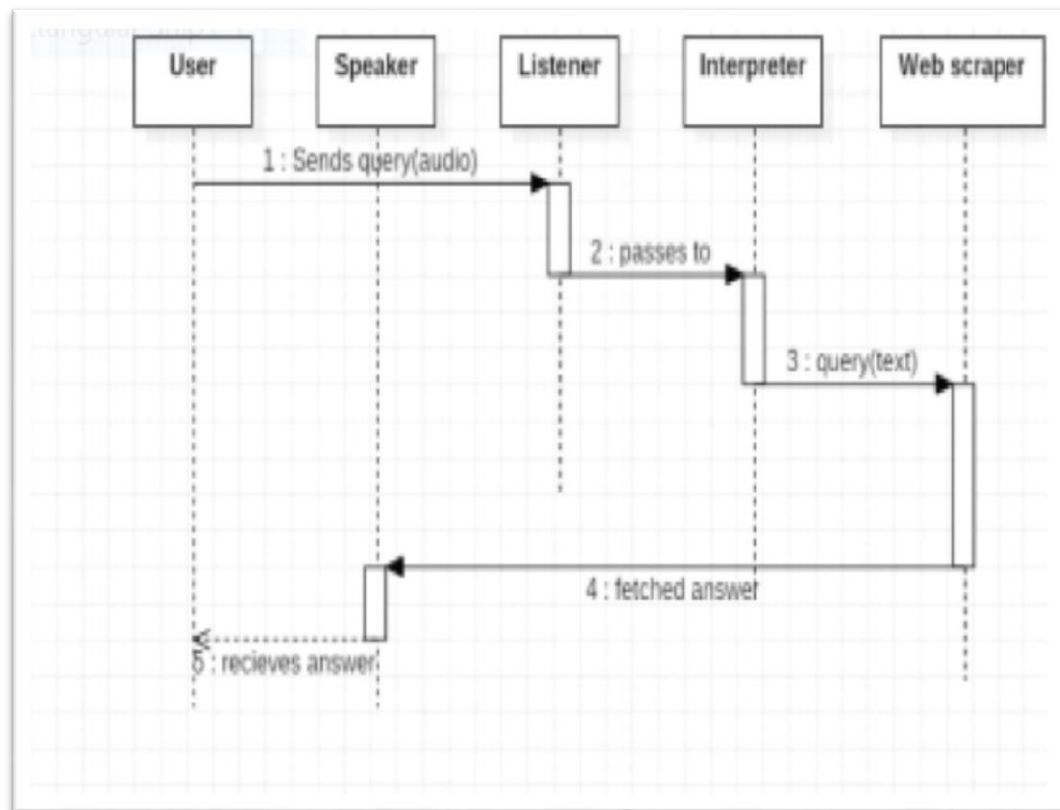


Fig11.1 Sequence Diagram for query-response

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. 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 executor. If the task is missing some information the virtual assistants ask user back about it. The received information is sent back to task and it is accomplished. After execution feedback is sent back to user.

## Chapter 4

# SYSTEM IMPLEMENTATION

---

### 4.1 Implementation Details

JARVIS, a desktop assistant is a voice assistant that can perform many daily tasks of desktop like playing music, opening your favourite IDE with the help of a single voice command. Jarvis is different from other traditional voice assistants in terms that it is specific to desktop and user does not need to make account to use this, it just require any internet connection while getting the instructions to perform any specific task.

As the first step, install all the necessary packages and libraries. The command used to install the libraries is “pip install” and then import it.

To install a package:

```
pip install <packagename>
```

Fig12.1 Python package installation

#### **FUNCTIONS USED:**

speak(): This function is used to make our JARVIS talk. It will take audio as an argument, and then it will pronounce it.

takeCommand(): The function is used to take the command as input through microphone of user and returns the output as string.

wishMe(): This function greets the user according to the time like Good Morning, Good Afternoon and Good Evening.

run\_jarvis(): This is the function which contains all the necessary task execution definition like “open google”, “open notepad”, “search on Wikipedia” ,”play music” and “open command prompt” etc.

**LIBRARIES AND PACKAGES USED:**

pyttsx3: A Python library that will help us to convert text to speech. In short, it is a text-to-speech library. It works offline, and it is compatible with python 2 as well as python 3.

SpeechRecognition: It is a python module which converts speech to text.

Datetime: This library provides us the actual date and time.

Wikipedia: It is a python module for searching anything on Wikipedia.

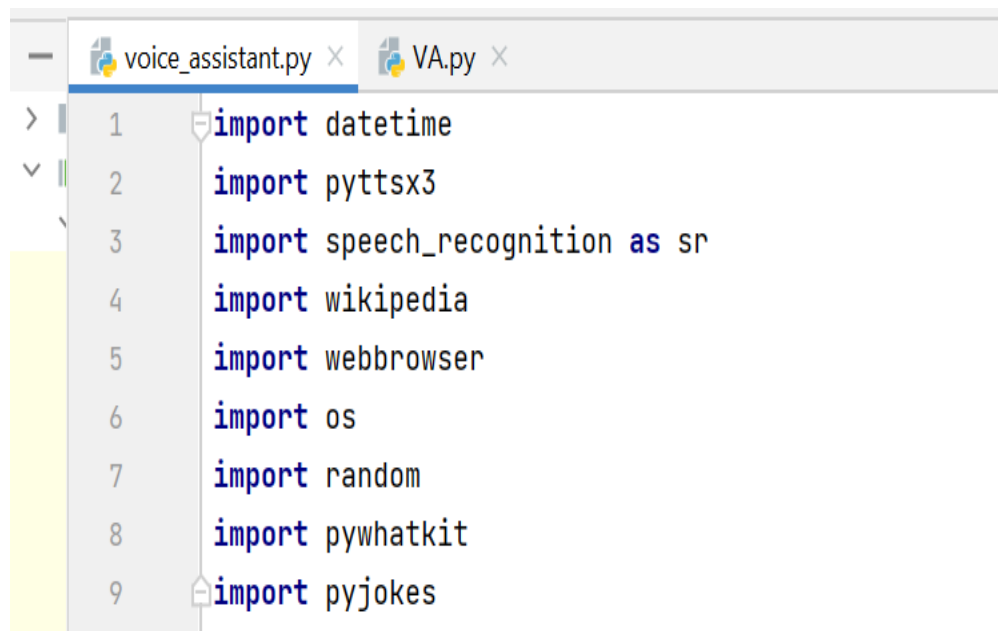
Pyjokes: It is a python libraries which contains lots of interesting jokes in it.

Webbrowser: It provides interface for displaying web-based documents to users.

os: It represents Operating System related functionality.

PyWhatKit: It is a Python Library which has number of features like playing YouTube videos, converting image to ASCII, sending emails etc.

Sapi5: The Speech Application Programming Interface or Sapi5 is an API developed by Microsoft to allow the use of speech recognition and speech synthesis within Windows applications. Applications that uses Sapi5 include Microsoft Office, Microsoft Speech Server.



```
1 import datetime
2 import pyttsx3
3 import speech_recognition as sr
4 import wikipedia
5 import webbrowser
6 import os
7 import random
8 import pywhatkit
9 import pyjokes
```

Fig13.1 Imported Modules

## 4.2 Coding

```

import datetime
import pyttsx3
import speech_recognition as sr
import wikipedia
import webbrowser
import os
import random
import pywhatkit
import pyjokes

#to use inbuilt voice from windows
#sapi5 is the Microsoft developed speech API which helps in synthesis and
recognition of voice
engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)
#voice id helps us to select different voices
def speak(audio):
    #to convert our text to speech
    engine.say(audio)
    engine.runAndWait() #without this command, speech will not be audible to
us.
def wishMe():
    #to greet the user
    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. Please tell me how may I help you')

def takeCommand():
    #it takes microphone input from the user and return 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')
        #using google for voice recognition
        print(f'User said: {query}\n')
        #user query will be printed
    except Exception as e:
        #print(e)
        print('Say that again please...')
        #say that again will be printed in case of improper voice
        return 'None'
        #none string will be returned
    return query

```

```

def run_jarvis():
    #converting user query into lower case
    query = takeCommand().lower()
    # logic for executing task based on query
    if 'wikipedia' in query:
        #if wikipedia found in the query then this block will be executed
        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:
        #if 'open youtube' found in the query then this block will be executed
        speak('What will you like to watch ?')
        pywhatkit.playonyt(f'{query}')
    elif 'open google' in query:
        #if 'open google' found in the query then this block will be executed
        webbrowser.open('google.com')
    elif 'open stackoverflow' in query:
        #if 'open stackoverflow' found in the query then this block will be
        executed
        webbrowser.open('stackoverflow.com')
    elif 'play music' in query:
        #if 'play music' found in the query then this block will be executed
        music_dir='D:\\Documents\\Music\\Arjit Singh1'
        songs = os.listdir(music_dir)
        print(songs)
        song = random.choice(songs)
        os.startfile(os.path.join(music_dir,song))
    elif 'the time' in query:
        #if 'the time' found in the query then this block will be executed
        strTime = datetime.datetime.now().strftime("%H:%M:%S")
        speak(f"Sir, the time is {strTime}")
    elif 'open command prompt' in query:
        #if 'open command' found in the query then this block will be executed
        os.system('start cmd')
    elif 'movie' in query:
        #if 'movie' found in the query then this block will be executed
        s=query.replace('movie', '')
        speak('playing'+s)
        pywhatkit.playonyt(s)
    elif 'joke' in query:
        #if 'joke' found in the query then this block will be executed
        a=pyjokes.get_joke()
        speak(a)
        print(a)
    elif 'search on youtube' in query:
        #if 'search on youtube' found in the query then this block will be
        executed
        query=query.replace('search on youtube', '')
        webbrowser.open(f'www.youtube.com/results?search_query={query}')

```

```

elif 'open notepad' in query:
    #if 'open notepad' found in the query then this block will be executed
    npath='C:\Windows\system32\notepad.exe'
    os.startfile(npath)
elif 'who created you' in query:
    #if 'who created you' found in the query then this block will be executed
    print(' I do not know their name, I created with python language,
in pycharm.')
    speak(' I do not know their name, I created with python language,
in pycharm.')
elif 'quit' in query:
    #if 'quit' found in the query then this block will be executed
    exit()
else:
    speak('Please say the command again..')

wishMe()
while True:
    run_jarvis()

```

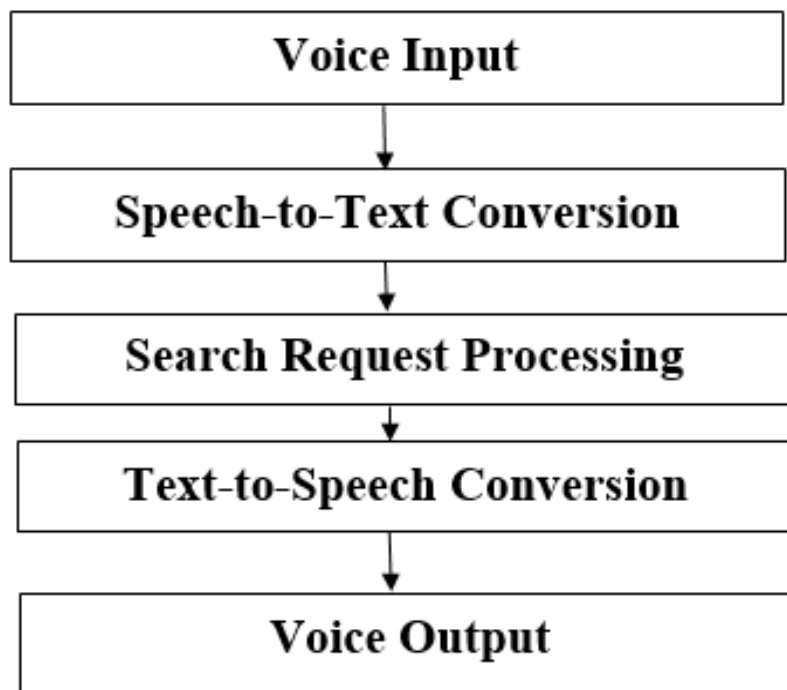


Fig14.1 Working

### 4.3 Snapshots



Fig15.1 Searching Wikipedia

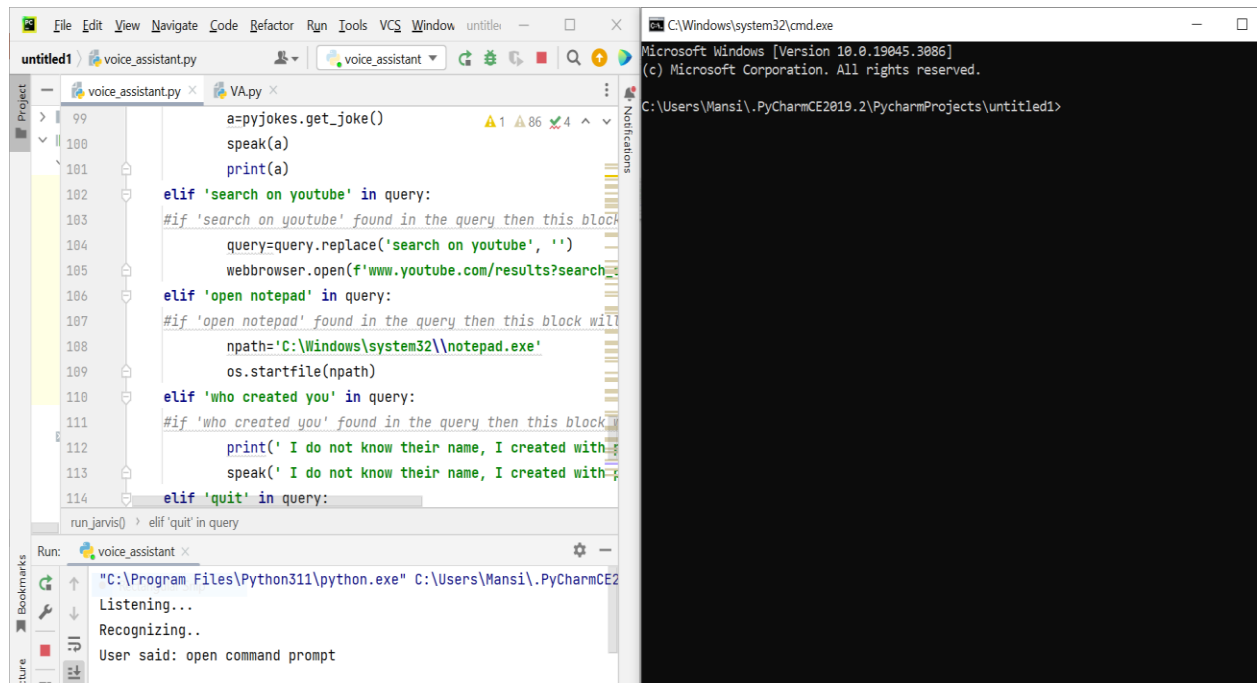


Fig15.2 Open Command Prompt



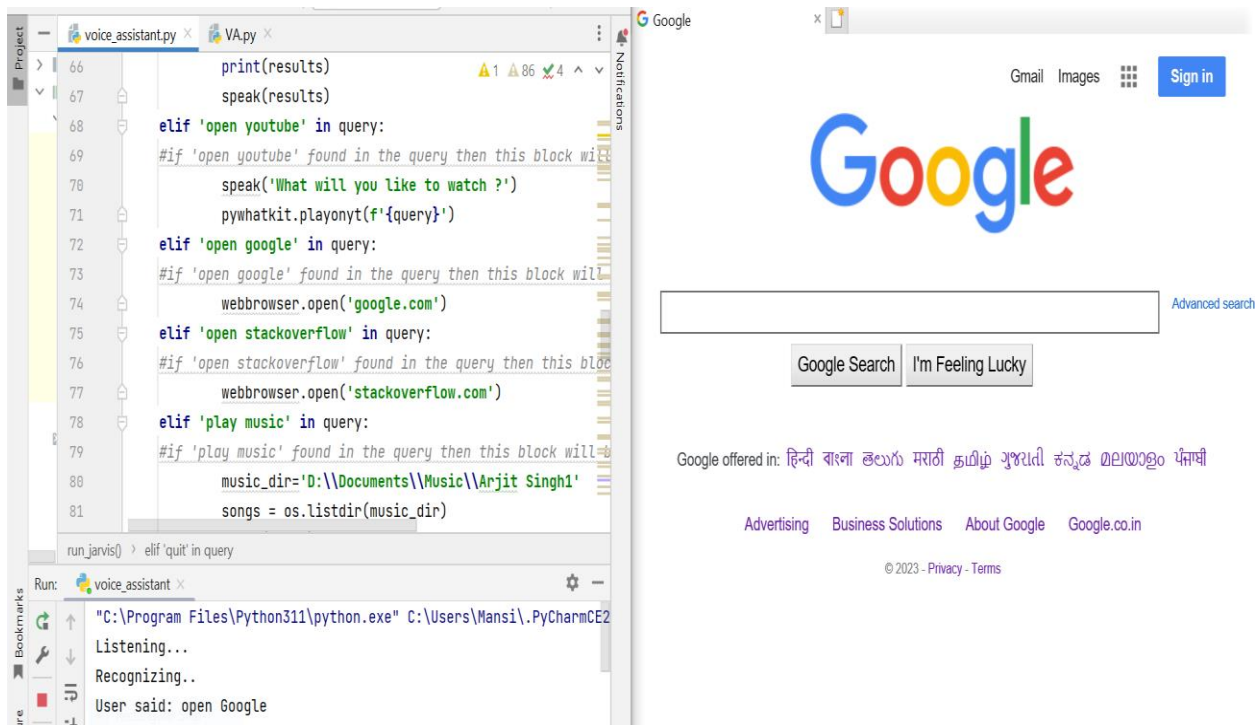


Fig15.3 Open Google

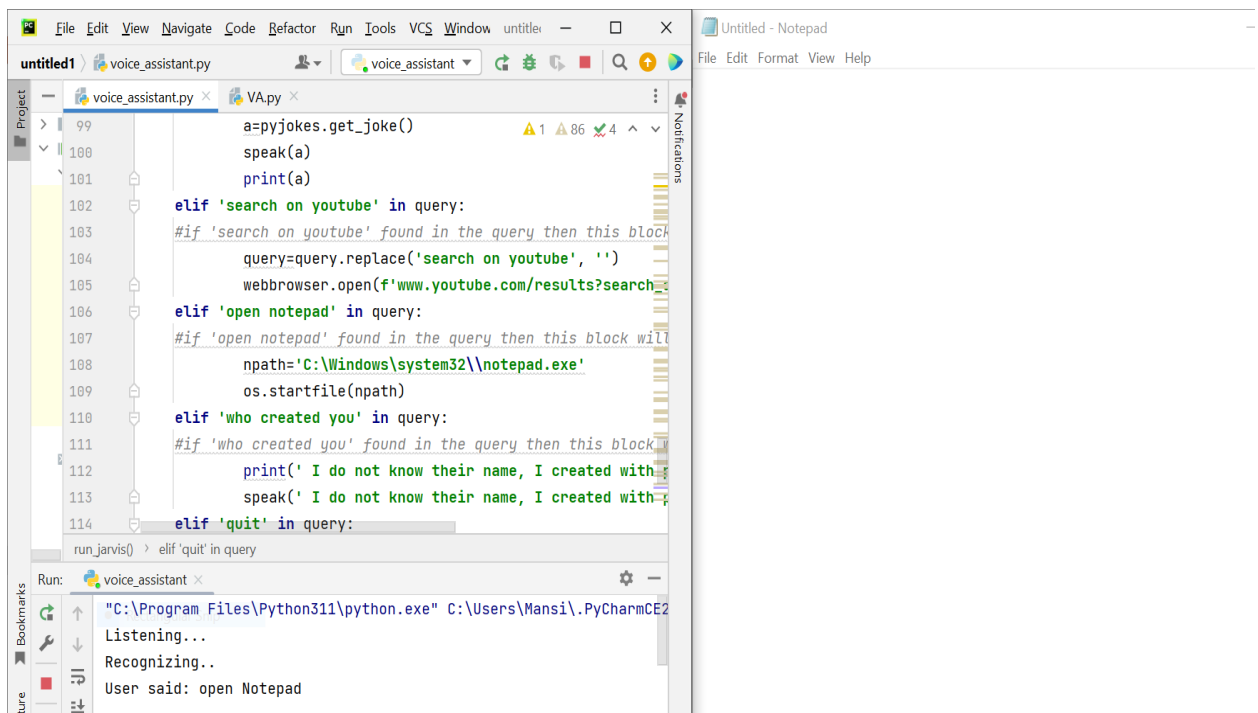


Fig15.4 Open Notepad

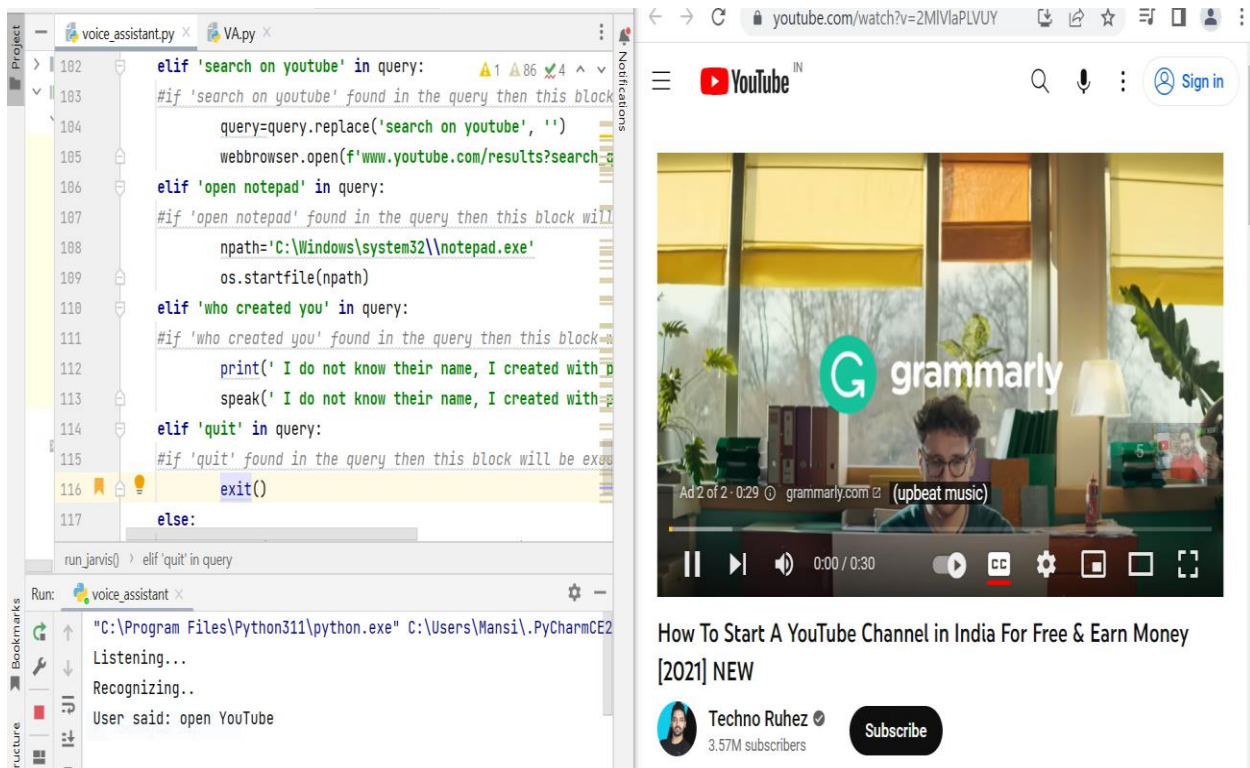


Fig15.5 Open Youtube

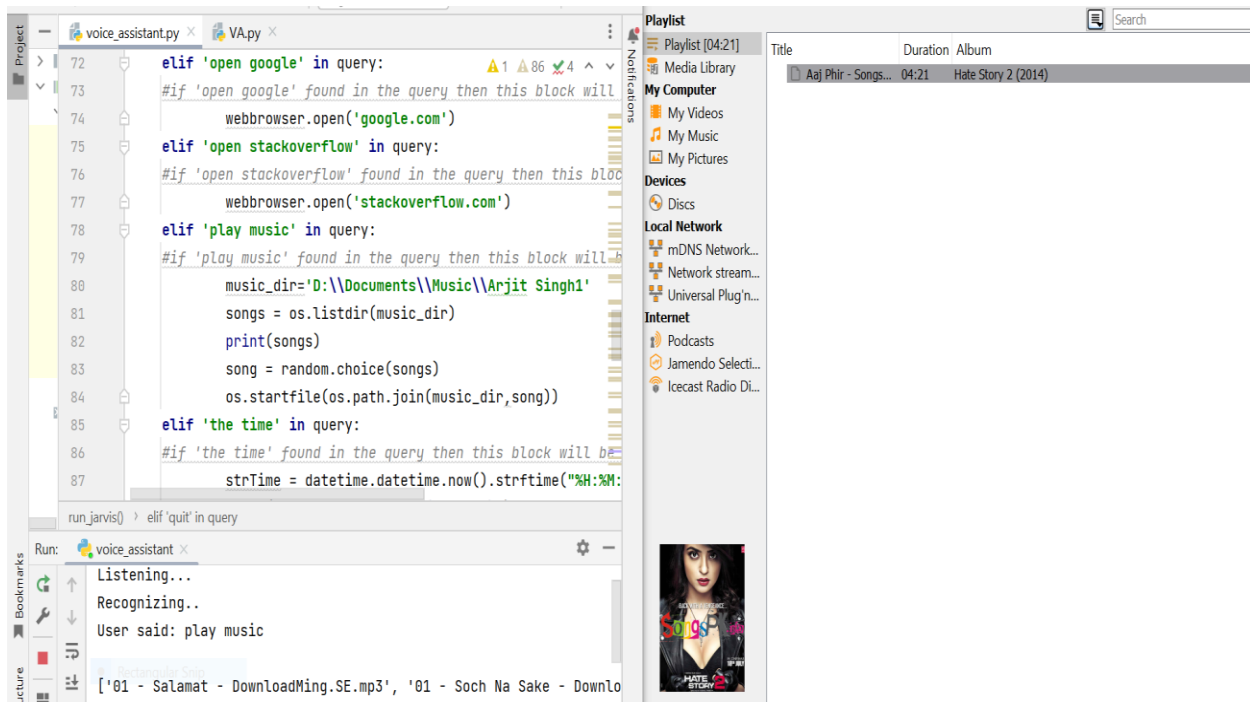


Fig15.6 Play music

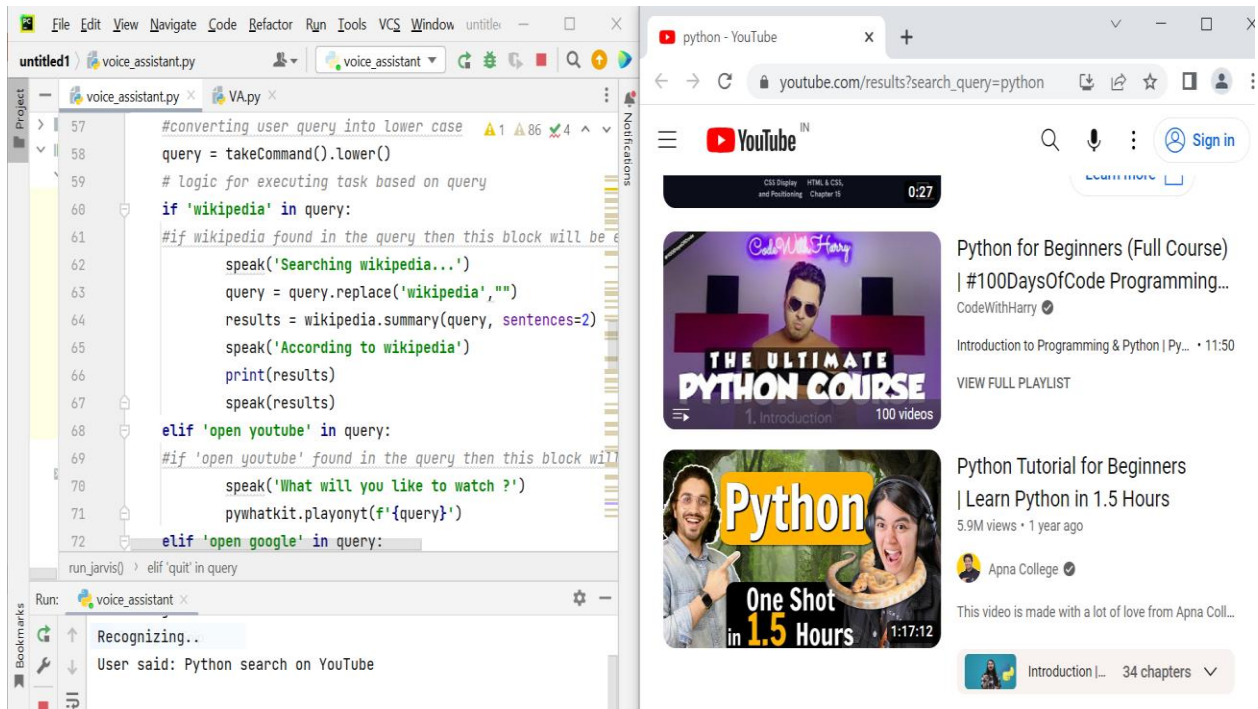


Fig15.7 Searching Python Tutorial on Youtube

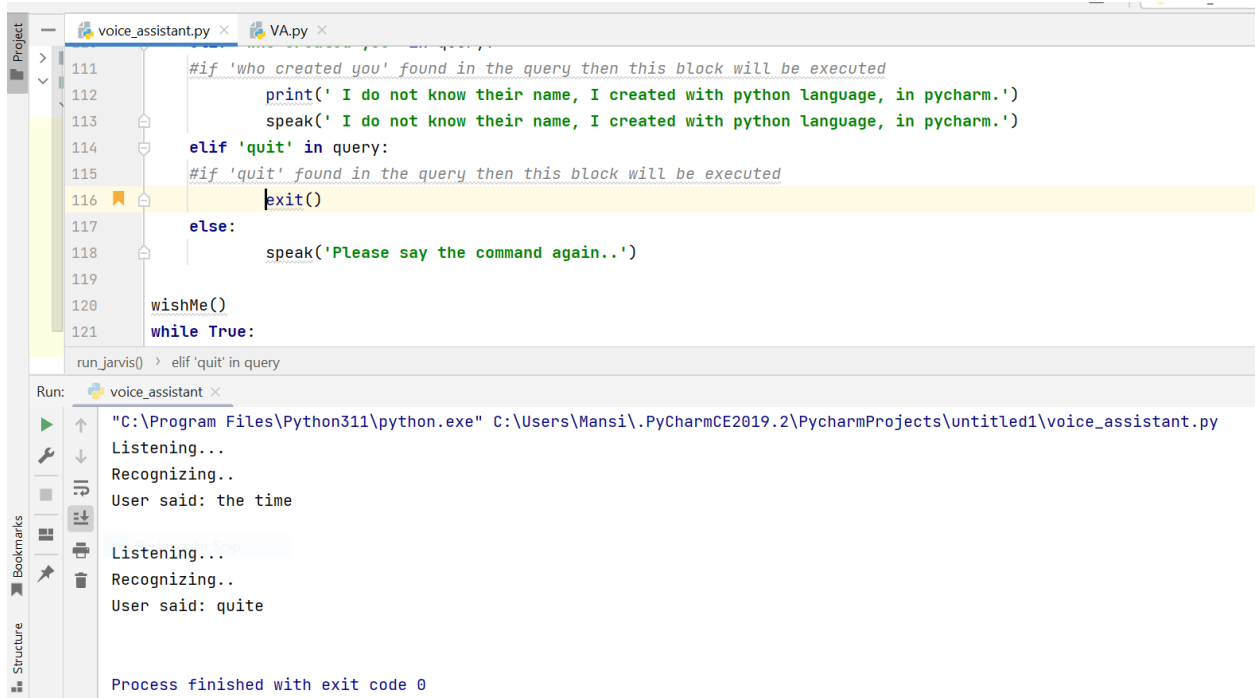


Fig15.8 Quit the assistant

## Chapter 5

### RESULT

#### 1.1 Comparative result to existing system

Sr. No.	Project	Technologies	Result	Issues
1.	Voice Assistant using python	Voice activation, automatic speech recognition, dialog management	Design and Implementation of digital assistance	Absence of additional or multiple features
2.	AI based voice assistant	Python 2.7, Spider, json, machine learning	A modern model with some advance features established	Similar with basic prototype and lacks multidimensionality
3.	An interpretation of AIML with integration of gTTS and Python	gTTS (Google text to speech), AIML (Artificial Intelligence Markup Language)	Integration of gTTS, AIML	Dependency on a particular platform
4.	Interoperability in virtual world	WWW(World Wide Web) services, HTTP, XML	Virtual world's communication, real world to virtual world (R2V)	Less vulnerable to modern operating systems
5.	Natural language understanding	Artificial Intelligence, Natural Language processing	Understanding of natural language processing, syntact processing	Only developing the understanding of NLP, difficult to a implement
6.	Chatbot song recommender system	Python, chatterbot library, list trainer	Developing basic chatbot system	Dedicated to a particular features only
7.	AI Chatbot in python	Pip, NumPy, tensorflow, random	Automated communication system developed	Limited to certain queries and conversation

## CONCLUSION

---

### CONCLUSION

JARVIS is a very helpful voice assistant without any doubt as it saves time of the user by conversational interactions, its effectiveness and efficiency. In this report we have discussed project details of the personal Desktop based voice assistant using Python which is built using open-source software PyCharm as an implementation tool. This Project will be helpful for people of all generations as well as to people with some disabilities or people with some special cases. The personal voice assistant will be easy to use and will reduce the manual human efforts for performing various tasks.

The functionality of the current voice assistant system is limited to working on Desktop based and working online (required to have internet connection to perform tasks) only. The voice assistant system is modular in nature so that addition of new features is possible without disturbing current system functionalities. Through this Desktop 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 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 gives responses to the user based on query being asked or the words spoken by the user. But while working on this project, there were some limitations encountered and also realized some scope of enhancement in the future which are mentioned below:

### LIMITATIONS

- Security is somewhere an issue, there is no voice command encryption in this project.
- Background voice can interfere
- Misinterpretation because of accents and may cause inaccurate results.
- JARVIS cannot be called externally anytime like other traditional assistants like Google Assistant can be called just by saying, “Ok Google!”

## **FUTURE SCOPE:**

- Make TUTOR learn on its own and acquire a new skill.
- Additionally, TUTOR android apps may also be made.
- Increase the number of voice terminals available.
- For further protection, voice instructions may be encrypted.

The severely disabled or those who have suffered minor repetitive stress injuries, i.e., those who may need the assistance of others to manage their surroundings. The use of an IVR system is increasing on a daily basis. Such technologies make it easier for the user to communicate with the computer system, which in turn facilitates the completion of a variety of activities.

The IVR system acts as an intermediary between humans and computers. Due to the time and research constraints, the existing IVR system is only suitable to desktop computers and will not be implemented in real phone devices. This is a disadvantage since the IVR system with Automatic Voice Recognition (AVR) may be used in a wide range of applications. Although the project is still in its infancy, there is plenty of room for improvement in the years to come.

The following are some of the places that may be relevant:

1. In Organizational inquiry desk the system may be utilised in different organisations for simple access to information about the organisation using the voice command.
2. Detection of isolated words is all that the suggested system does, but it might be expanded to include audio to text conversion with additional improvements in algorithms.
3. It is possible to employ voice recognition of Nepali phrases in order to accomplish the work of freshly generated apps and therefore create more user pleasant applications.
- 4 In embedded systems, voice commands may be used to handle multiple activities using speech recognition technology. This promotes automation of labour and can thus be very advantageous in industrial process automation process automation.

## REFERENCES

---

1. <https://www.studocu.com>
2. <https://www.scribd.com/document/525812125/Virtual-Assistant-Project-REPORT>
3. <https://www.studocu.com/in/document/manonmaniam-sundaranar-university/computer-graphics-and-visualization/project-report-on-virtual-assistant-subm/26288529>
4. <https://pdfcoffee.com>
5. <https://en.wikipedia.org/>
6. <https://ijcrt.org/papers/IJCRT22A6338.pdf>
7. <https://www.studocu.com/in/document/b-k-birla-college/marketing/jarvis-project-report/19701174>



Fig16.1 Jarvis