

“VIRTUAL VOICE ASSISTANT”



A PROJECT REPORT

Submitted in partial fulfillment towards the requirement of the degree of

BACHELOR OF COMPUTER APPLICATION

(2019-2022)

Guided by:

Dr. Deepti Verma

Submitted by:

Suman Bhattacharya

Vishal Choudhary

SHRI VAISHNAV INSTITUTE OF MANAGEMENT, INDORE

Affiliated to Devi Ahilya Vishwavidyalaya, Indore, Madhya Pradesh

Approval

The project entitled **“Virtual Voice Assistant”** submitted by **Suman Bhattacharya (90400581)** and **Vishal Choudhary (90400588)** of **BCA VI Semester** is approved for the partial fulfillment towards the requirement of degree of **Bachelor of Computer Application**.

Internal Examiner
Head, DOCS

External Examiner
Director

Acknowledgement

We would like to take this opportunity to record our deep sense of gratitude to all those who helped us in achieving this target.

First and foremost, we would like to express our gratitude towards **Dr. George Thomas, Director, Shri Vaishnav Institute of Management, Indore** for extending all the facilities needed to carry out this project.

It was proud enough for us to simply be awarded a project under the able guidance of our guide Mrs. Deepti Verma. She was present all along the work, with their ideas, inspiration and encouragement, and provided a masterly all through our work.

We are also grateful to Dr.Kshama Paithankar, HOD, Department of Computer Science and Dr. Jitendra Jain, Programme Coordinator, BCA for extending all the facilities needed to carry out this project & reviewing the entire part of it with great attention.

We extend our thanks to all other staff members of the Department of Computer Science.

Suman Bhattacharya

Vishal Choudhary

INDEX

S.no.	Contents	Page no.
1.	Problem Investigation 1.1 Introduction 1.2 Goal and Objectives	1-3
2.	System Analysis Information Gathering Feasibility Study Data Flow Diagram Hardware and Software Requirements	4-18
3.	Tool Used Front End Back End	19-21
4.	Testing	22-23
5.	Implementation Forms Layout	24-39
6.	Conclusion Salient Features of System Limitations of System Scope of Future Enhancements	40-43
7.	Bibliography	44

INTRODUCTION

In today's era almost all tasks are digitized. We have Smartphones in our hands and it is nothing less than having the world at your fingertips. These days we aren't even using fingers. We just speak of 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 Assistant. It also supports specialized tasks such as booking a flight, or finding the cheapest book online from various e-commerce sites and then providing an interface to book an order, helping automate search, discovery and online order operations.

Virtual Assistants are software programs that help you ease your day to day tasks, such as showing a weather report, creating reminders, making shopping lists etc. They can take commands via text (online chat bots) or by voice. Voice based intelligent assistants need an invoking word or wake word to activate the listener, followed by the command. For my project the wake word is Suman. We have so many virtual assistants, such as Apple's Siri, Amazon's Alexa and Microsoft's Cortana. For this project, wake-word was chosen as Suman.

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. Suman is effortless to use. Call the wake word 'Suman' followed by the command. And within seconds, it gets executed.

Voice searches have dominated over text search. Web searches conducted via mobile devices have only just overtaken those carried out using a computer and the analysts are already predicting that 50% of searches will be via voice by 2020. Virtual assistants are turning out to be smarter than ever. Allow your intelligent assistant to make email work for you. Detect intent, pick out important information, automate processes, and deliver personalized responses. 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.

BACKGROUND

There already exist a number of desktop virtual assistants. A few examples of current virtual assistants available in the market are discussed in this section along with the tasks they can provide and their drawbacks.

SIRI from Apple

SIRI is personal assistant software that interfaces with the user through voice interface, recognizes commands and acts on them. It learns to adapt to the user's speech and thus improves voice recognition over time. It also tries to converse with the user when it does not identify the user request. It integrates with calendar, contacts and music library applications on the device and also integrates with GPS and camera on the device. It uses location, temporal, social and task based contexts, to personalize the agent behavior specifically to the user at a given point of time.

ALEXA from Amazon

Alexa is Amazon's voice-based AI-powered digital assistant that powers an entire smart device ecosystem. Alexa can reply to simple queries and perform various tasks or commands that one gives.

Supported Tasks

1. Launch an application on my Computer
2. Send a text message to someone
3. Send email to someone
4. Play song on youtube
5. Weather updates
6. Read any mail or text

OBJECTIVES

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 may be done in a business environment, for example, on the business website, with a chat interface. On the mobile platform, the intelligent virtual 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.

Virtual assistants can tremendously save you time. We spend hours in online research and then making the report in our terms of understanding. Suman can do that for you. Provide a topic for research and continue with your tasks while Suman does the research. Another difficult task is to remember test dates, birthdays or anniversaries. It comes with a surprise when you enter the class and realize it is a class test today. Just tell Suman 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 time¹⁵. In this respect, the ability of personal assistants to accurately recognize spoken words is a prerequisite for them to be adopted by consumers.

SYSTEM ANALYSIS

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. The complete analysis is followed below.

Problem definition

Usually, a 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 a need for a system that can manage tasks effortlessly.

We already have multiple virtual assistants. But we hardly use it. There are a number of people who have issues in voice recognition. These systems can understand English phrases but they fail to recognize our accent. Our way of pronunciation is way distinct from theirs. Also, they are easier to use on mobile devices than desktop systems. There is a need for a virtual assistant that can understand English in an Indian accent and work on a desktop system.

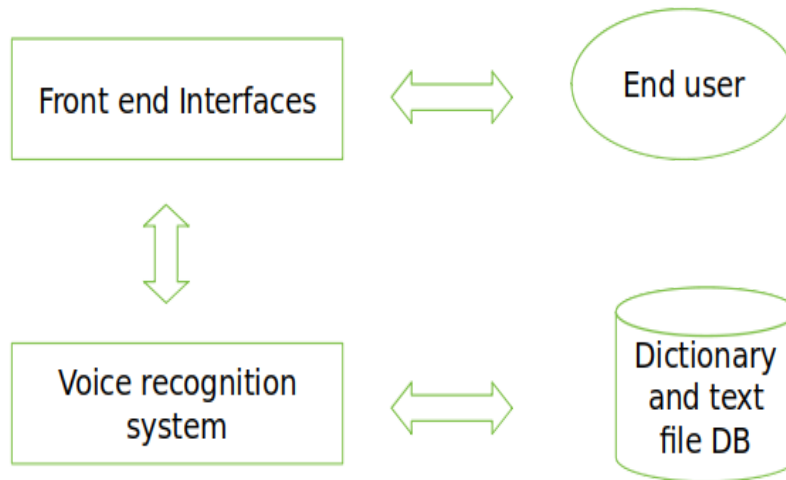
When a virtual assistant is not able to answer questions accurately, it's because it lacks the proper context or doesn't understand the intent of the question. Its ability to answer questions relevantly only happens with rigorous optimization, involving both humans and machine learning. Continuously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

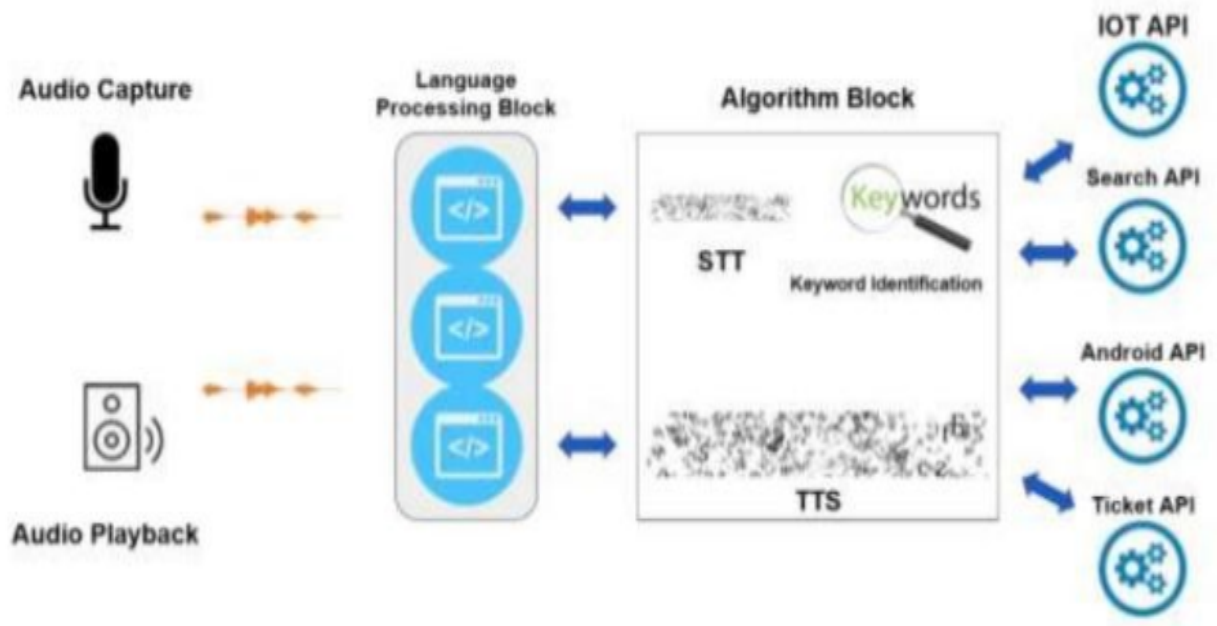
Virtual Assistant should be able to model complex task dependencies and use these models to recommend optimized plans for the user. It needs to be tested for finding optimum paths when a task has multiple sub-tasks and each sub-task can have its own sub-tasks. In such a case there can be multiple solutions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

Structured systems analysis is a specific technique for systems analysis that covers all activities from initial understanding of the problem through to specification and high-level design of the software system.

So in our case

Considering overall research, voice applications will be used in the following three ways: Firstly, command to the computer whereas secondly, to input information to the computer, finally for communication with other people. In this section we will be discussing general components for voice application





System Architecture The total design consists of these phases:

- 1) Collection of data which is in speech format.
- 2) Analyse the voice and convert it to text.
- 3) storing the data and processing it.

Speech generation from the text output that is processed

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 the 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 a virtual assistant, the user must have a microphone to convey their message and a speaker to listen when the system speaks. These are very cheap nowadays and everyone generally possesses them. Besides, the system needs internet connection. While using SUMAN, 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.

2. **Operational feasibility:** It is the ease and simplicity of operation of the 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 how to write can read out problems for the system and get answers.

3. **Economical feasibility:** Here, we find the total cost and benefit of the proposed system over the current system. For this project, the main cost is documentation cost. Users also would have to pay for a microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, Suman won't cost too much.

4. **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.

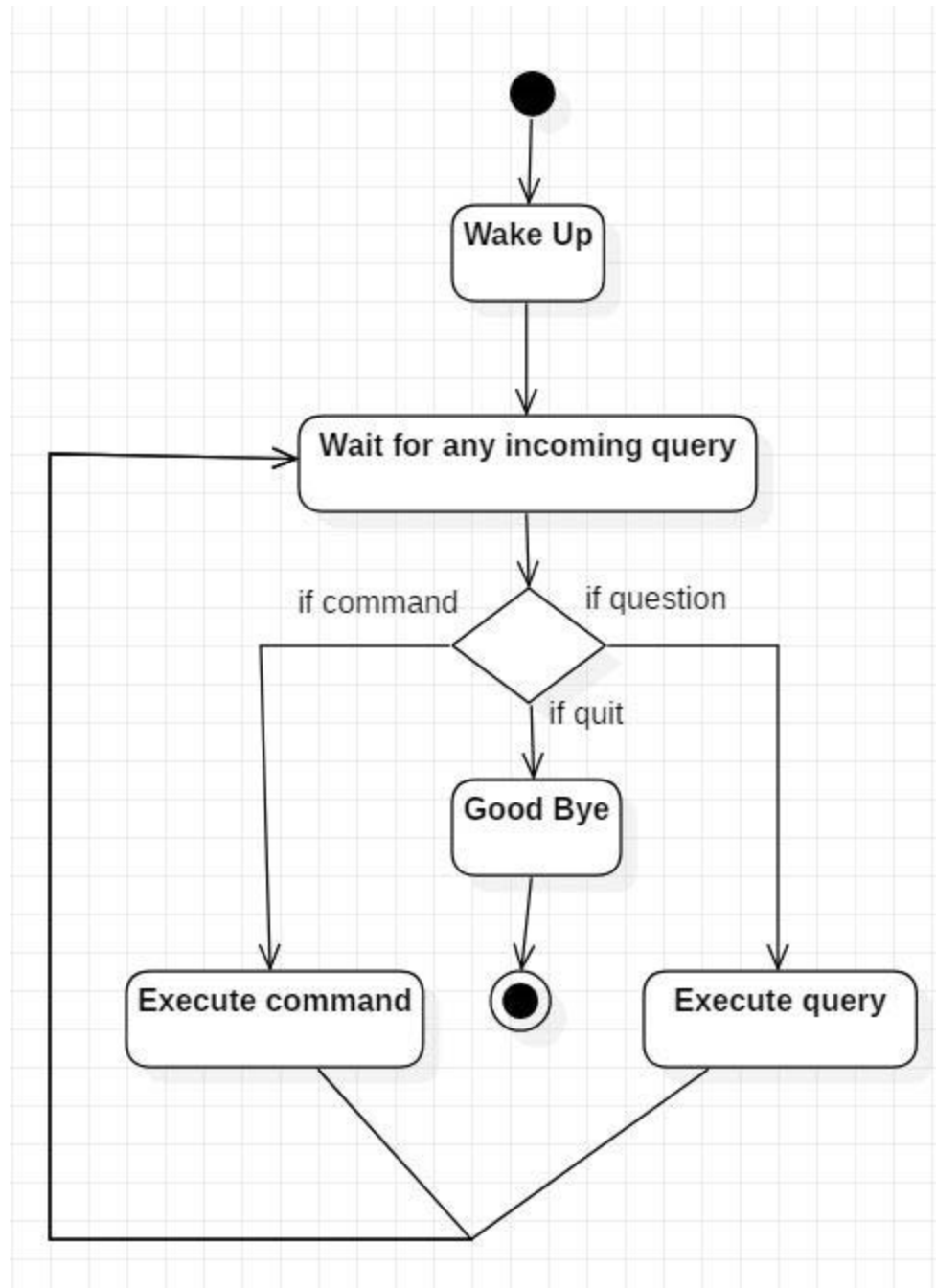
5. **Cultural feasibility:** It deals with compatibility of the project with the cultural environment. Virtual assistant is built in accordance with the general culture. The project is named Suman so as to represent Indian culture without undermining local beliefs.

This project is technically feasible with no external hardware requirements. Also it is simple in operation and does not cost training or repairs. Overall feasibility study of the project reveals that the goals of the proposed system are achievable. Decision is taken to proceed with the project.

DATA FLOW DIAGRAM

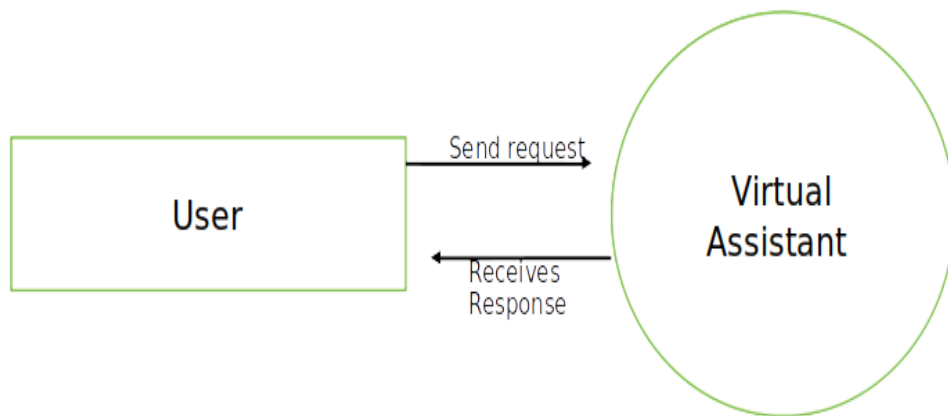
Initially, the system is in idle mode. As it receives any wake up call it begins execution.

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

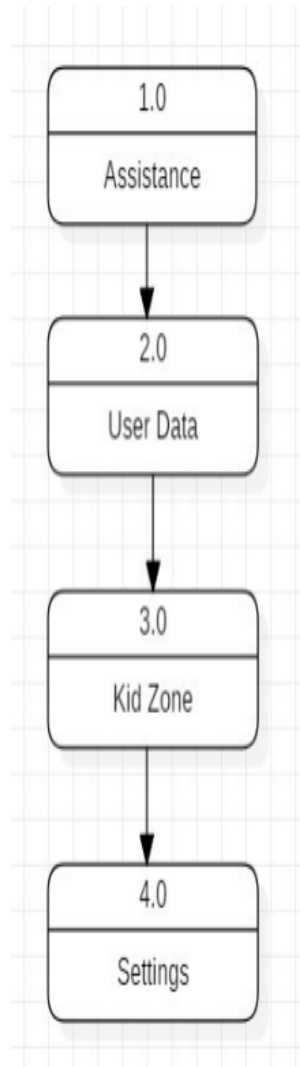


Activity Diagram

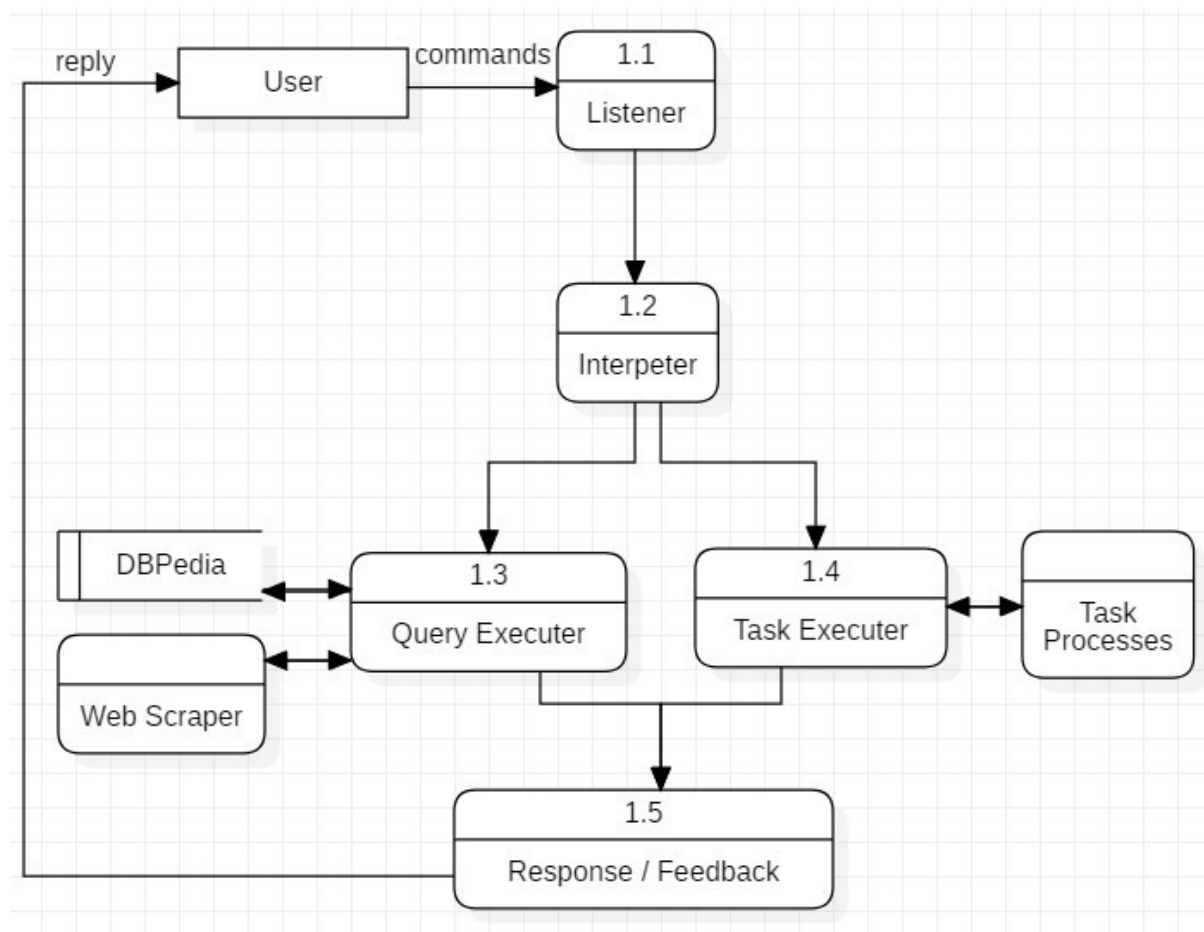
DATA FLOW DIAGRAM



DFD Level 0

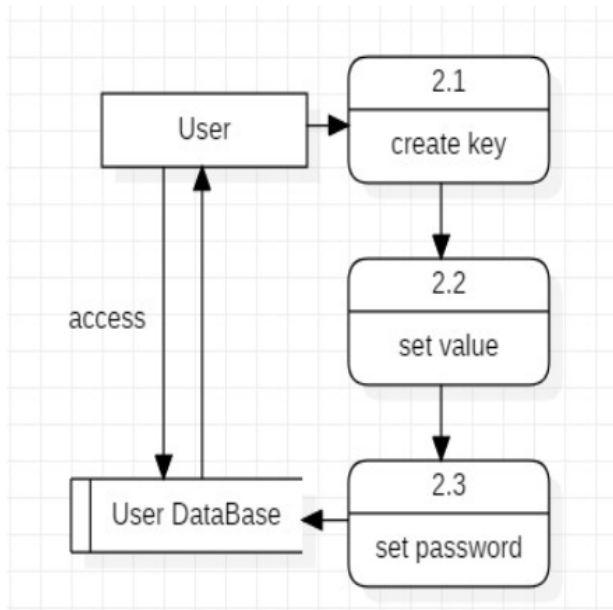


DFD Level 1

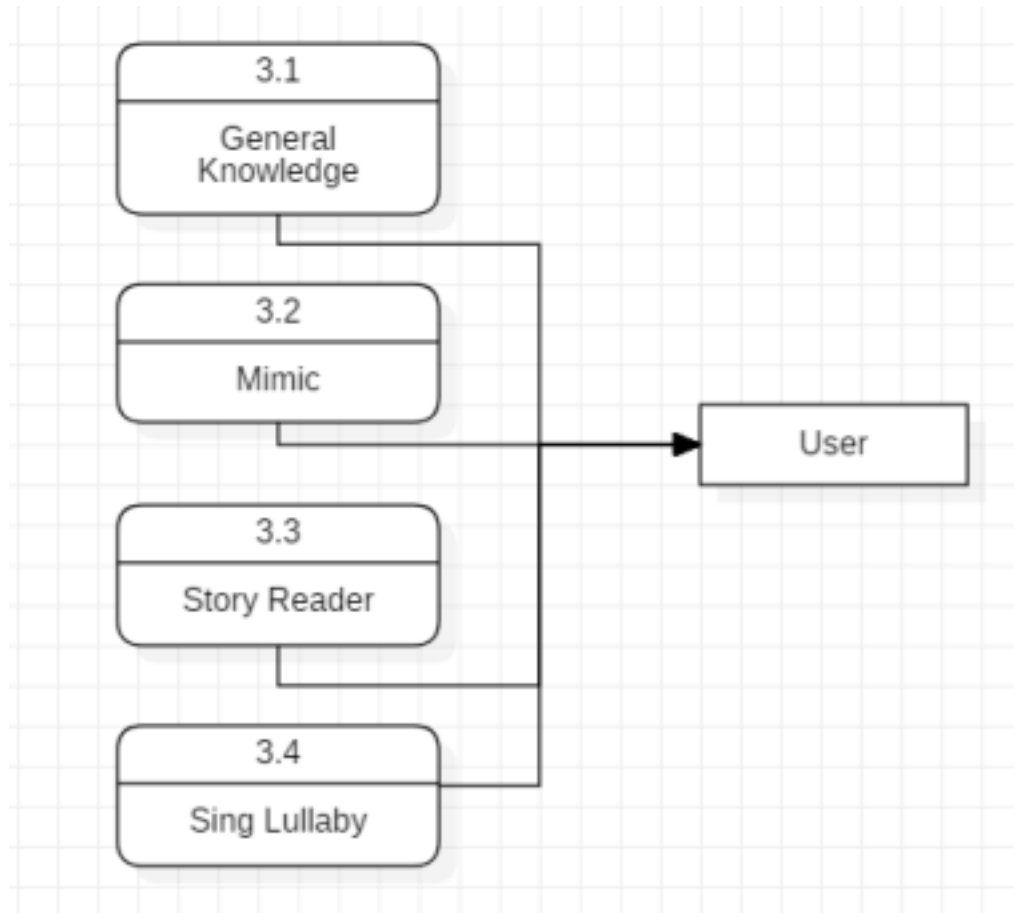
DFD Level 2

Data flow in assistant

DFD Level 2

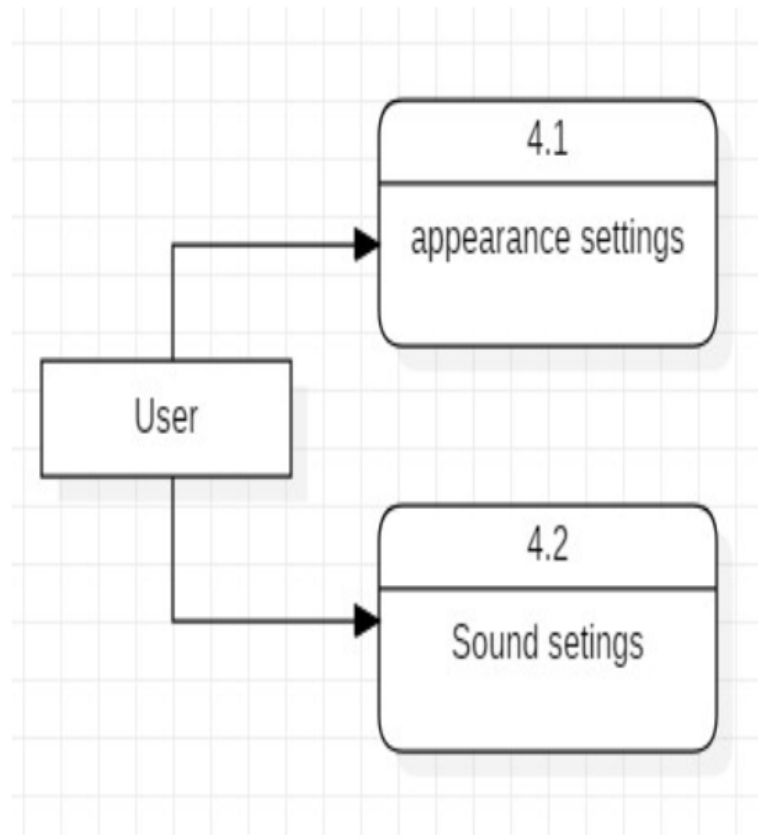


Managing user Data



Data Flow in Kid Zone

DFD Level 2

**Settings of virtual Assistant**

HARDWARE AND SOFTWARE REQUIREMENTS

The software is designed to be light-weighted so that it doesn't be a burden on the machine running it. This system is being built keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirements for virtual assistants.

Hardware:

- Pentium-pro processor or later.
- RAM 512MB or more.

Software:

- Windows 7(32-bit) or above/Linux
- Python 2.7 or later
- Pycharm

TOOL USED

Front end

Not necessarily needed, If we use any IDE or code editor that is enough for us. But definitely we can have a front end like a dialog box or a window type front end , and yet it still works fine.

Back end

Python

Python is an OOPs (Object Oriented Programming) based, high-level, interpreted programming language. It is a robust, highly useful language focused on rapid application development (RAD). Python helps in easy writing and execution of codes. Python can implement the same logic with as much as 1/5th code as compared to other OOPs languages.

Python provides a huge list of benefits to all. The usage of Python is such that it cannot be limited to only one activity. Its growing popularity has allowed it to enter into some of the most popular and complex processes like Artificial Intelligence (AI), Machine Learning (ML), natural language processing, data science etc. Python has a lot of libraries for every need of this project. For Suman, libraries used are speech recognition to recognize voice, Pyttsx for text to speech, selenium for web automation etc.

PyCharm

PyCharm is an integrated development environment used in computer programming, specifically for the Python programming language. It is developed by the Czech company JetBrains.

Pyttsx3

Pyttsx3 stands for Python 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 Mac OS X, Windows, and Linux. It works for both Python2.x and 3.x versions. Its main advantage is that it works offline.

Speech Recognition

This is a library for performing speech recognition, with support for several engines and APIs, online and offline. It supports APIs like Google Cloud Speech API, IBM Speech to Text, Microsoft Bing Voice Recognition etc.

Smtplib

The smtplib is a Python library for sending emails using the Simple Mail Transfer Protocol (SMTP). The smtplib is a built-in module; we do not need to install it. It abstracts away all the complexities of SMTP.

PyAutoGUI

PyAutoGUI is a cross-platform GUI automation Python module for human beings. Used to programmatically control the mouse & keyboard.

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.

pywhatkit

pywhatkit is a python module for sending WhatsApp messages at a certain time. To install the pywhatkit module, Type the following command in your IDE/Compiler: pip install pywhatkit. This command will download the pywhatkit module. It will cause some delay as it will download some related modules too

NLTK Tokenizer Package

The Natural Language Toolkit (NLTK) is a Python package for natural language processing. NLTK requires Python 3.7, 3.8, 3.9 or 3.10.

Tokenizer is a compact pure-Python (≥ 3.6) executable program and module for tokenizing Icelandic text. It converts input text to streams of tokens, where each token is a separate word, punctuation sign, number/amount, date, e-mail, URL/URI, etc. It also segments the token stream into sentences, considering corner cases such as abbreviations and dates in the middle of sentences.

pyjokes

One line jokes for programmers (jokes as a service). Once installed, simply call `pyjoke` from the command line or add it to your `.bashrc` file to see a joke every time you call it.

TESTING

Test Case 1

Test Title: Response Time

Test ID: T1

Test Priority: High

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

Description:

Time is very critical in a voice based system. As we are not typing inputs, we are speaking them. The system must also reply in a moment. Users must get an instant response to the query made.

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 an answer in a moment is of no use if the answer is not correct. Accuracy is of utmost importance in a virtual assistant system.

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 an approximate value. For example, if someone asks for value of PI the system must respond with an approximate value and not the accurate value. Getting exact values in such cases is undesirable.

NOTE : There might include a few more test cases and these test cases are also subject to change with the final software development.

Implementation

So after done with some line of codes written below. Its working.

```
import random

import pyttsx3
import datetime
import speech_recognition as sr
import pyaudio
import smtplib
from secret import senderemail, epwd, email_list , user_name
from email.message import EmailMessage
import pyautogui
import webbrowser as wb
from time import sleep
import wikipedia
import pywhatkit
import requests
from newsapi import NewsApiClient
import clipboard
import os
import subprocess
import pyjokes
import psutil
from nltk.tokenize import word_tokenize
```

```
engine = pyttsx3.init()
engine.setProperty('rate', 180)
```

```
engine.say("Hello ")
engine.runAndWait()
```

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

```
def getvoices(voice):
    voices = engine.getProperty('voices')

    if voice == 1:
        engine.setProperty('voice', voices[38].id)

    if voice == 2:
        engine.setProperty('voice', voices[29].id)
```

```
def time():
    Time = datetime.datetime.now().strftime("%I:%M:%S")
    speak('time is ')
    speak(Time)
```

```
def date():
```

```

year = int(datetime.datetime.now().year)
month = int(datetime.datetime.now().month)
date = int(datetime.datetime.now().day)
speak("date is")
speak(date)
speak(month)
speak(year)

```

```

def greeting():
    hour = datetime.datetime.now().hour
    if hour >= 6 and hour < 12:
        speak("good morning sir")
    elif hour >= 12 and hour < 18:
        speak("good afternoon sir")
    elif hour >= 18 and hour < 24:
        speak("good evening sir")
    else:
        speak("good night sir")

```

```

def takeCommandCMD():
    query = input("please tell me how can help you?")
    return query

```

```

def takeCommandMIC():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Adjusting for background noise. One second")
        r.adjust_for_ambient_noise(source)

```

```

    print("Listening.....")
    r.pause_threshold = 1
    audio = r.listen(source)

    try:
        print("recognizing...")
        query = r.recognize_google(audio, language="en-IN")
        print(query)
    except Exception as e:
        print(e)
        speak("say that again sir")
        return "None"
    return query

```

```

def sendEmail(receiver, subject, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login(senderemail, epwd)
    email = EmailMessage()
    email['From'] = senderemail
    email['To'] = receiver
    email['Subject'] = subject
    email.set_content(content)
    server.send_message(email)
    server.close()

```

```

def sendwhatsappmsg(phone_no, message):
    Message = message
    wb.open('https://web.whatsapp.com/send?phone='+phone_no+'&text='+Message)

```

```

sleep(10)
pyautogui.press('enter')

def serachgoogle():
    speak('what should I search for?')

    search = takeCommandMIC()
    wb.open('https://www.google.com/search?q='+search)

def news():
    newsapi = NewsApiClient(api_key='42b62fd20fef4b51b843604201a517fd')
    data = newsapi.get_top_headlines(q='bitcoin',
                                     language='en',
                                     page_size=5)
    newsdata = data['articles']
    for x,y in enumerate(newsdata):
        print(f'{x} {y["description"]}')
        speak(f'{x} {y["description"]}')

    speak("that's it for now i'll update you in while")

def text2speech():
    text = clipboard.paste()
    print(text)
    speak(text)

def screenshot():
    nam_img = (datetime.datetime.now())
    #nam_img = '\\fallen1\\Downloads'
```



```
img = pyautogui.screenshot(nam_img)
img.show()
```

```
def flip():
    speak("okay sir, flipping a coin")
    coin = ['head', 'tail']
    toss = []
    toss.extend(coin)
    random.shuffle(toss)
    toss = ("".join(toss[0]))
    speak("i flipped a coin and you got a "+toss)
```

```
def cpu():
    usage = str(psutil.cpu_percent())
    speak("CPU is at" + usage)
    battery = psutil.sensors_battery()
    speak("Battery is at" + battery)
    speak(battery.percent)
```

```
if __name__ == '__main__':
    getvoices(2)
    greeting()
    wakeword = "suman"
```

```

while True:
    query = takeCommandMIC().lower()
    query= word_tokenize(query)
    print(query)
    if wakeword in query:

        if 'time' in query:
            time()

        elif 'date' in query:
            date()

        elif 'email' in query:

            try:
                speak("To whome you want to send")
                name = takeCommandMIC().lower()
                receiver = email_list[name]
                speak("what is the subject of mail")
                subject = takeCommandMIC()
                speak('what should i say?')
                content = takeCommandMIC()
                sendEmail(receiver, subject, content)
                speak("email has been sent")
            except Exception as e:
                print(e)
                speak("unable to send")

```

elif 'offline' in query:

```
    speak("this is Suman signing off from duty")
    quit()
```

elif 'whatsapp' in query:

```
    try:
        speak("To whome you want to WhatsApp ")
        name = takeCommandMIC().lower()
        phone_no = user_name[name]
        speak("what is the message?")
        message = takeCommandMIC().lower()
        sendwhatsappmsg(phone_no, message)

        speak("message has been sent")
    except Exception as e:
        print(e)
        speak("unable to send")
```

elif 'wikipedia' in query:

```
    speak('searching on wikipedia...')
    query=query.replace("wikipedia", "")
    result = wikipedia.summary(query, sentences = 2)
    print(result)
    speak(result)
```

elif 'google' in query:

```
searchgoogle()
```

```
elif 'youtube' in query:
```

```
    speak('what you wanna see on youtube?')
    topic = takeCommandMIC()
    pywhatkit.playonyt(topic)
```

```
elif 'weather' in query:
```

```
    url =
'http://api.openweathermap.org/data/2.5/weather?q=indore&units=imperial&appid=c02ad853
2b20192c4ae557854406a9ba'
    res = requests.get(url)
    data = res.json()

    weather = data['weather'] [0] ['main']
    temp = data['main']['temp']
    desp = data['weather'][0] ['description']
    temp = round((temp - 32)* 5/9)
    print(weather)
    print(temp)
    print(desp)
    speak('Temperature : {} deegrees celcius'.format(temp))
    speak('weather is {}'.format(desp))
```

```
elif 'news' in query:
```

```
    news()
```

elif 'read' in query:

text2speech()

elif 'covid' in query:

covid()

elif 'open' in query:

speak("what you want to open?")

#codepath = '/usr/share/applications/spyder3.desktop'

app = takeCommandMIC().lower()

subprocess.call(app)

#elif 'documents' in query:

os.popen("cd /home/fallen1/path ; subl")

elif 'joke' in query:

speak(pyjokes.get_joke())

elif 'screenshot' in query:

screenshot()

elif 'remember' in query:

speak("what should i remember?")

```

data = takeCommandMIC()
speak("you said to me "+data)
remember = open('data.txt','w')
remember.write(data)
remember.close()

```

```

elif 'flip' in query:
    flip()

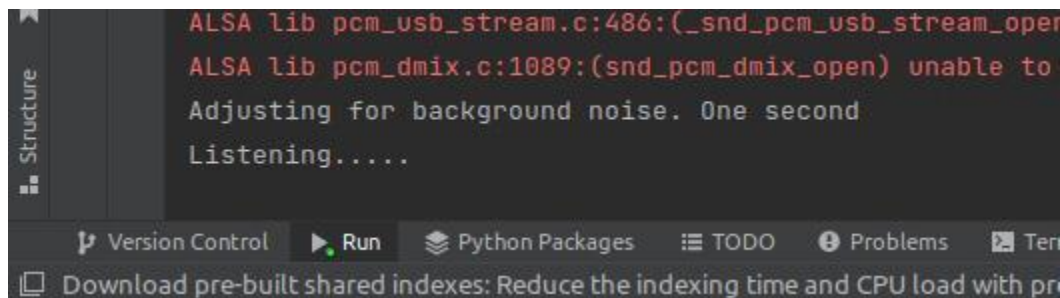
```

```

elif 'cpu' in query:
    cpu()

```

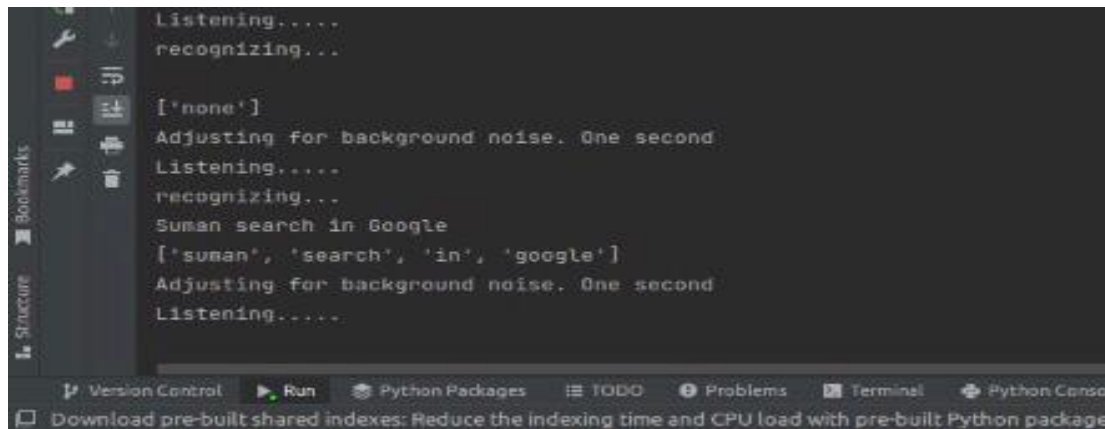
So when we hit run :-



It's started listening and waits for the wake word and keywords.

Now for example we say Suman search on Google.(Suman= Wake word, search on google = keyword for google search)

Then it will be started recognizing



```
Listening.....
recognizing...

['none']
Adjusting for background noise. One second
Listening.....
recognizing...
Suman search in Google
['suman', 'search', 'in', 'google']
Adjusting for background noise. One second
Listening.....
```

And as it finds both wake and keyword in the query it will reply to us “what you want to search for?” and it will start listening again.

Now for example we say “Vaishnav Institute of Management “ then it will again recognize it

The screenshot shows a PyCharm IDE with a project named 'ass' located at '/media/fallen1/OS/Users/bismoo/ass'. The main file is 'main.py'. The code in 'main.py' is as follows:

```

10 import pyautogui
11 import webbrowser as wb
12 from time import sleep
13 import wikipedia
14 import pywhatkit
15 import requests
16 from newsapi import NewsApiClient
17 import clipboard
18 import os
19 import subprocess
20 import pyjokes
21 import psutil
22 from nltk.tokenize import word_tokenize
23
24
25
26 engine = pyttsx3.init()
27 engine.setProperty('rate', 180)
28
29 engine.say("Hello ")
30 engine.runAndWait()
31
32
33 def speak(audio):
34     engine.say(audio)
35     if __name__ == '__main__':
36         while True:
37             if wakeword

```

The console output shows the following sequence of events:

```

Run: main
recognizing...
['none!']
Adjusting for background noise. One second
Listening.....
recognizing...
Suman search in Google
['suman', 'search', 'in', 'google']
Adjusting for background noise. One second
Listening.....
recognizing...
Vaishnav Institute of Management
Adjusting for background noise. One second

```

When it's done with recognizing it will open a new window with our default browser and show us the result and in the background waiting for the next query.

The screenshot shows a web browser window with a search for "Vaishnav Institute of Management". The search results page displays the institute's name, location (Indore, Madhya Pradesh), and a table of course admissions. The browser's address bar shows the search query, and the top of the page features a navigation bar with links to "Overview" and "Course admissions".

Search Results:

Vaishnav Institute of Management

About 14,80,000 results (0.52 seconds)

Shri Vaishnav Institute of Management, Indore
College in Indore, Madhya Pradesh

[Overview](#) [Course admissions](#)

<https://www.svimi.org>

Shri Vaishnav Institute of Management:: Home
Shri Vaishnav Institute of Management, Indore | Management and Computer Science Institute
| MBA, MCA, BBA, BCA, B.Sc. (CS / IT).

Placement
Shri Vaishnav Institute of Management, Indore ...

Contact Us
Shri Vaishnav Institute of Management, Indore ...

[More results from svimi.org »](#)

Course admissions >

Common streams	Courses
Management and Business Administration	6 >
Computer Applications and IT	4 >
Sciences	2 >
Media, Mass Communication, and Journalism	1 >

About

[svimi.org](https://www.svimi.org)

Address: Sector B, Gumasta Nagar, Scheme 71, Indore, Madhya Pradesh 452002

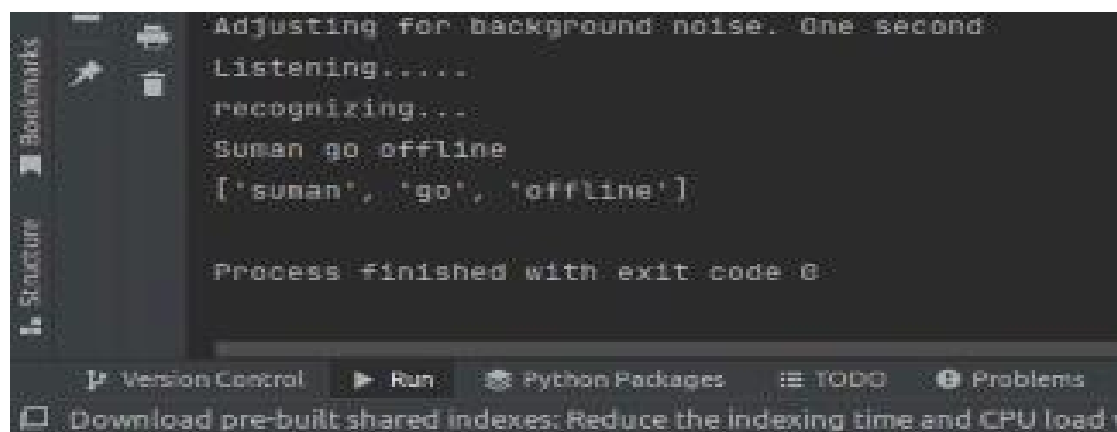
Phone: 0731 278 0011

People also search for

- [Devi Ahilya Vishwa...](#)
- [Shri Vaishnav Vidyap...](#)
- [Prestige Institute of Man...](#)
- [Sanghvi Institute of Man...](#)

[See more](#)

And now we give another command , query or question, or we can give a quit command as in our case its “Offline” then it will quit the program.

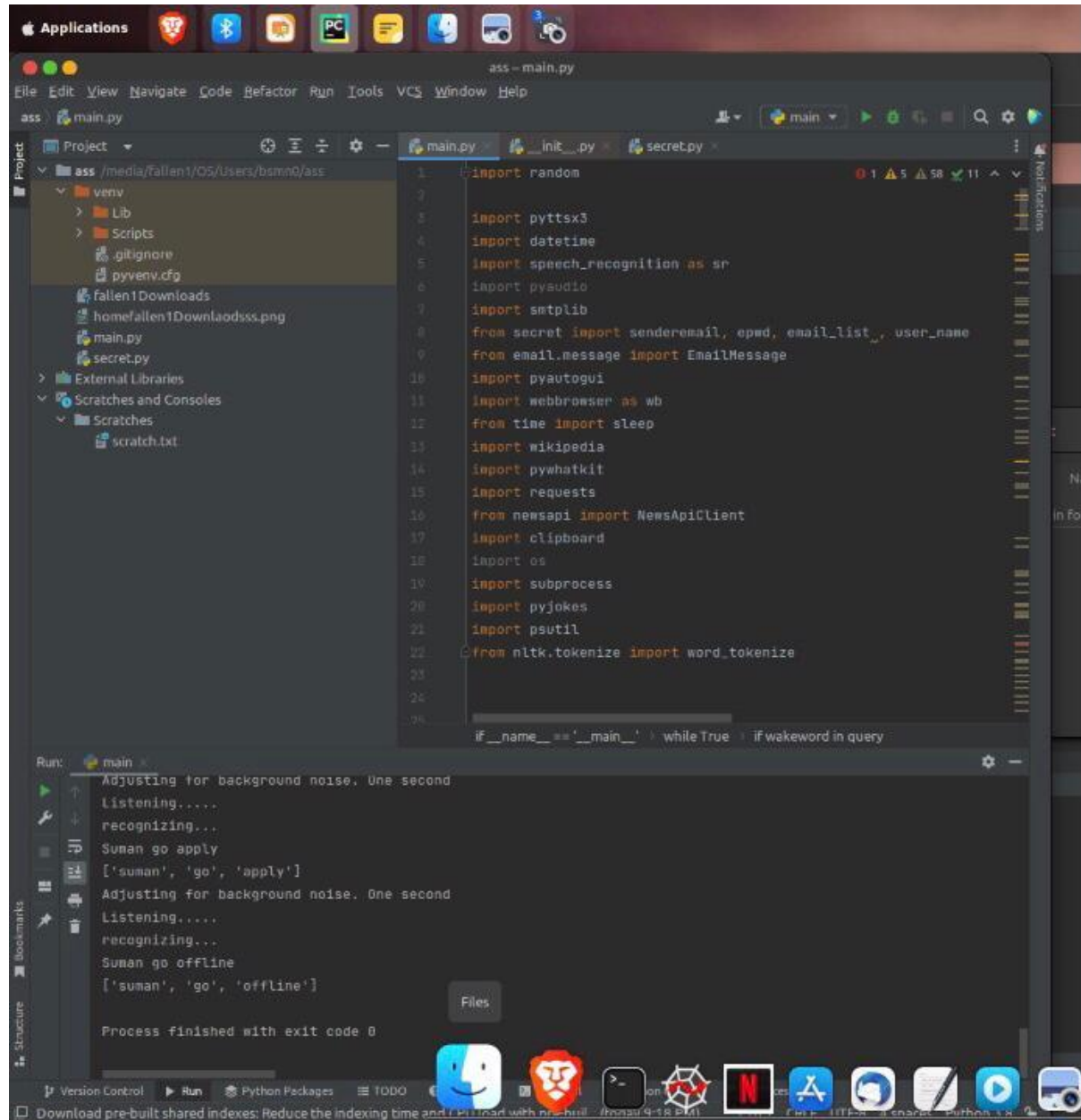


```
Adjusting for background noise. One second  
Listening.....  
recognizing...  
Suman go offline  
['suman', 'go', 'offline']  
  
Process finished with exit code 0
```

The screenshot shows a terminal window within an IDE. The left sidebar contains 'Structure' and 'Bookmarks' tabs. The terminal output shows the process of adjusting for background noise, listening, and recognizing the command 'Suman go offline'. The recognized words are listed as ['suman', 'go', 'offline']. The process concludes with 'Process finished with exit code 0'. The IDE's bottom status bar includes tabs for 'Version Control', 'Run', 'Python Packages', 'TODO', and 'Problems', along with a checkbox for 'Download pre-built shared indexes: Reduce the indexing time and CPU load'.

As it's again found the wake word and keyword for quitting it will exit with code 0 .
That means everything is working perfectly.

Overall Layout



CONCLUSION

Salient Features of System

Hands free access

Hands free access , that means we don't have to use our hands all the time. Just by voice commands we can perform tasks.

Speech to Text/NLTK

All our speech is converted to text so that the assistant can get keywords easily. NLTK is a Natural Language Toolkit which provides us with some special features like Tokenizer. Tokenizer basically tokenizes any sentence to separate words.

Automate Tasks

Automating tasks is something that we all wanted. Imagine we want to send a mail to a friend so normally we have to go to Browser and any mail client then login with our email id then click on new mail then we have to fill with details ,like email id then subject and then details and then only we are able to send it. But here We just have to tell “send email” then it will ask us “whom you want send email?” then we just have to give the name. It will take email from a secret file or dictionary database . Then it will ask for the Subject of mail then “ what should i say?” Then we just have to speak to the body .We want it will automatically send it and notify us with “Email has been sent”.

Answer Questions

Simply answer any type of Questions like to search any Person , place or thing and like weather updates and many more.

Limitations of System

Lack of Accuracy and Misinterpretation

Voice recognition software won't always put your words on the screen completely accurately. Programs cannot understand the context of language the way that humans can, leading to errors that are often due to misinterpretation. When you talk to people, they decode what you say and give it a meaning. Voice recognition software can do this but may not be capable of choosing the correct meaning. For example, it cannot always differentiate between homonyms, such as "their" and "there." It may also have problems with slang, technical words and acronyms.

Accents and Speech Recognition

Voice recognition systems can have problems with accents. Even though some may learn to decode your speech over time, you have to learn to talk consistently and clearly at all times to minimize errors. If you mumble, talk too fast or run words into each other, the software will not always be able to cope. Programs may also have problems recognizing speech as normal if your voice changes, say when you have a cold, cough, sinus or throat problem.

Background Noise Interference

To get the best out of voice recognition software, you need a quiet environment. Systems don't work so well if there is a lot of background noise. They may not be able to differentiate between your speech, other people talking and other ambient noise, leading to transcription mix-ups and errors. This can cause problems if you work in a busy office or noisy environment. Wearing close-talking microphones or noise-canceling headsets can help the system focus on your speech.

But we added a background noise cancellation to avoid initial level noises.

Scope of Future Enhancements

A voice virtual assistant can help people in the future to automate more day-to-day tasks, such as scheduling and organizing their lives. The accuracy of the devices will likely increase if we use machine learning to categorize results, and then use those categories in further queries. The accuracy of the devices is increasing exponentially in recent years. In future development, a voice virtual assistant could be designed to accept commands in bilingual language and respond back in the same language queried by the user.

Well from here we can develop it in as many ways as we want, we have two modules or two plans that we will start working on soon and we can implement in future.

We can develop it to help visually impaired people or old people who are losing their eyesight day by day. Simply, we have to design it for mobile and by accessing mobile's inbuilt camera can help them. We will train our machine with things, people's etc and speak to whatever it sees with a mobile camera so if there is a bottle in front of the camera it will speak like a water bottle. It's just a small example but we can help them with many more things. If we got succeeded definitely we will be launching it for free.

Here is our secondly business model so in this case we are going to train our machine (AI) with a set of weapons, to recognize humans and then recognize weapons and whenever it sees any weapon it hits an alarm or something. And we can approach this model to banks, ATMs, hospitals and public places etc. Mainly in the ATM and bank it will help a lot.

These two plans are going to be implemented soon but till we can do many things with our Virtual voice assistant (AI).

Combining machine learning with natural language processing could create a powerful, human-like machine that can honestly interact with us and respond to our needs.

BIBLIOGRAPHY

Websites referred

1. www.stackoverflow.com
2. www.pythonprogramming.net
3. www.codecademy.com
4. www.tutorialspoint.com
5. www.google.co.in

Udemy -

Learn To Create Advance AI Assistant (JARVIS 2.0) With Python

THANK YOU

