SPEECH RECOGNITION SYSTEM

A major project report submitted in the partial fulfillment of the requirement for the award of the Degree

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Submitted by

ASHISH KUMAR – 17IT0385
DEEPSHIKHA SHARMA – 17IT0386
UMA BHARTI – 17IT0387

Under the Guidance of

Mr. Vikash Kumar Agarwal
(Assistant Professor)

DEPARTMENT OF INFORMATION TECHNOLOGY
RTC INSTITUTE OF TECHNOLOGY
ANANDI, ORMANJHI, RANCHI, JHARKHAND-835219

2017-2021

SPEECH RECOGNITION SYSTEM

A major project report submitted in the partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Submitted by

ASHISH KUMAR – 17IT0385
DEEPSHIKHA SHARMA – 17IT0386
UMA BHARTI – 17IT0387

Under the Guidance of

Mr. Vikash Kumar Agarwal
(Assistant Professor)

DEPARTMENT OF INFORMATION TECHNOLOGY
RTC INSTITUTE OF TECHNOLOGY
ANANDI, ORMANJHI, RANCHI, JHARKHAND-835219
2017-2021

DECALARATION

We hereby declare that the work described in this major project, entitled **SPEECH RECOGNITION SYSTEM** which is been submitted by us in partial fulfillment for the award of degree of Bachelor of Technology in **Information Technology** to the **Ranchi University, Ranchi** is a record of Bonafide work carried out by us under the guidance of **Mr. Vikash Kumar Agarwal**

The results embodied in this major project report have not been submitted to any other University or Institute for the award of any degree.

Place:Ranchi
Date:

Ashish kumar (17IT0385)
Deepshikha sharma (17IT0386)
Uma Bharti(17IT0387)

CERTIFICATE

This is to certify that the major project report entitled **SPEECH RECOGNITION SYSTEM** being submitted by Ashish kumar, Deepshikha sharma, Uma Bharti in the partial fulfillment for the award of the degree of Bachelor of Technology in **Information Technology** to Ranchi University, Ranchi is a record of bonafied work carried out by him/her under my guidance and supervision.

The result embodied in this major project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Mr. Vikash Kumar Agarwal **External**

Guide
Department of CSE
RTC Institute of Technology
Anandi, Ormanjhi, Ranchi, 835219

Mr. RAJESH SINGH

HOD
Department of IT
RTC Institute of Technology
Anandi, Ormanjhi, Ranchi, 835219

ACKNOWLDGEMENT

This project report would not have been written without the support and encouragement of all those people whom I want to thank today. Firstly we would like to express our sincere gratitude to **Mr. Rajesh Singh**, HOD, Department of Computer Science & Engineering for giving an opportunity to do this project. We would like to thank our guide **Mr. Vikash Kumar Agarwal** , Department of Computer Science & Engineering whose role as project guide was in valuable for the project. We extremely thankful for the keen interest he took in advising us for book and references provided for the moral support extended to us.

We convey our gratitude to all our facilities for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the cordial support they offered.

Last but not the least a great thanks to our parents who play the most important role in making us the person we are today. Without their unconditional love, support, encouragement and sacrifices, we would never have been able to achieve today. This has always been their dream and I am grateful to god for giving me the opportunity to fulfill their wishes.

Date:

ABSTRACT

In the Modern Era of fast moving technology we can do things which we never thought we could do before but, to achieve and accomplish these thoughts there is a need for a platform which can automate all our tasks with ease and comfort. Thus we need to develop a Personal Assistant having brilliant powers of deduction and the ability to interact with the surroundings just by one of the materialistic form of human interaction i.e. HUMAN SPEECH. The Hardware device captures the audio request through microphone and processes the request so that the device can respond to the individual using in-built speaker module. For Example, if you ask the device ’what’s the weather?’ or ’how’s traffic?’ using its built-in skills, it looks up the weather and traffic status respectively and then returns the response to the customer through connected speaker.

CONTENTS

List of Figures

1.Introduction

- Speech recognition is the process of converting an acoustic signal, captured by a microphone or a speaker, to set of words.
- The recognised words can be end in themselves, as for application such as command and control, data entry and document presentations.
- They can also serves as the input to further linguistics processing in order to achieve speech understanding.

.

CLASSIFICATION OF SPEECH RECOGNITION SYSTEM

Speech recognition system can be classified in several different types by describing the type of speech utterance, type of speaker model and type of Vocability that they have the ability to recognize, The challenges are briefly explained below -

Types of speech utterance :-

Isolated word**:** Isolated word recognizer usually requires each spoken word to have quiet (lack of an audio signal) on both side of the sample window. It accepts single word at a time.

Connected word **:** It is similar to isolated word, but it allow separate utterances to 'run together' which contains a minimum pause in between them.

Continuous speech **:** It allows the users to speak naturally and in parallel the computer will determine the content.

Spontaneous speech : It is the type of speech which is natural sounding and is not rehearsed.

Types of speaker model :-

Speech recognition system is broadly into two main categories based on speaker models namely speaker dependent and speaker independent.

Speaker dependent models (SDM) **:** These systems are designed for a specific speaker. They are easier to develop and more accurate but they are not so flexible.

Speaker independent models (SIM): These systems are designed for variety of speaker. These systems are difficult to develop and less accurate but they are very much flexible.

Types of Vocabulary : **-**

The Vocabulary size of speech recognition system affects the processing requirement, accuracy and complexity of the system. In voice recognition system: speech- to-text the types of vocabularies can be classified as follows :

Small Vocabulary**:** single letter
Medium Vocabulary : two or three letter words
Large Vocabulary: more letter words

2. Hardware / Software Requirement

2.1 Hardware Requirement

- 32-bit or 64-bit architecture.
- Processor must be of 1.6GHz
- 4 GB RAM
- Monitor
- Microphone

2.2 Software Requirement

- Python IDLE (3.7)
- Text Editor (PYCHARM) Edition 2021.1.1

3. Software Requirement Analysis

3.1 Define The Problem:

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 Speech Recognition System. But we hardly use it. There are number of people who have issues in speech 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 speech recognition system that can understand English in Indian accent and work on desktop system.

When a speech recognition system 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 speech recognition system learning undesired bad behaviors. They require large amount of information to be fed in order for it to work efficiently.

Speech recognition system 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.

3.2 Modules And Their Functionalities:

- Pyttsx3:- This module is used for the conversion of text to speech in a program it works offline. To install this module type the below command in the terminal.

Syntax - pip install pyttsx3

•Speech Recognition:- Since we are building an application of voice assistant, one of the most important things in this is that your assistant recognizes your voice(means what you want to say).

Syntax - pip install SpeechRecognition

•PyAudio:- PyAudio is a set of python binding for portaudio, cross-platform library interfacing with audio drivers.

In pycharm to install PyAudio, Firstly we have to install pipwin

Syntax - pip install pipwin

Then Install PyAudio

Syntax - pipwin install PyAudio

Speech Input Using a Microphone and Translation of Speech to Text

- Allow Adjusting for Ambient Noise: Since the surrounding noise varies, we must allow the program a second or too to adjust the energy threshold of recording so it is adjusted according to the external noise level.
- Speech to text translation: This is done with the help of Google Speech Recognition. This requires an active internet connection to work. However, there are certain offline Recognition systems such as PocketSphinx, but have a very rigorous installation process that requires several dependencies. Google Speech Recognition is one of the easiest to use.

Translation of Speech to Text:

First, we need to import the library and then initialize it using init() function. This function may take 2 arguments.
init(driverName string, debug bool)

- drivername: [Name of available driver] sapi5 on Windows | nsss on MacOS
- debug: to enable or disable debug output

After initialization, we will make the program speak the text using say() function.
This method may also take 2 arguments.
say(text unicode, name string)

- text: Any text you wish to hear.
- name: To set a name for this speech. (optional)

Finally, to run the speech we use runAndWait() All the say() texts won’t be said unless the interpreter encounters runAndWait().

4. Software Design

4.1 PROJECT MODULES

The work started with analyzing the audio commands given by the user through the microphone. This can be anything like getting any information, operating a computer's internal files, etc. This is an empirical qualitative study, based on reading above mentioned literature and testing their examples. Tests are made by programming according to books and online resources, with the explicit goal to find best practices and a more advanced understanding of Speech Recognition System.

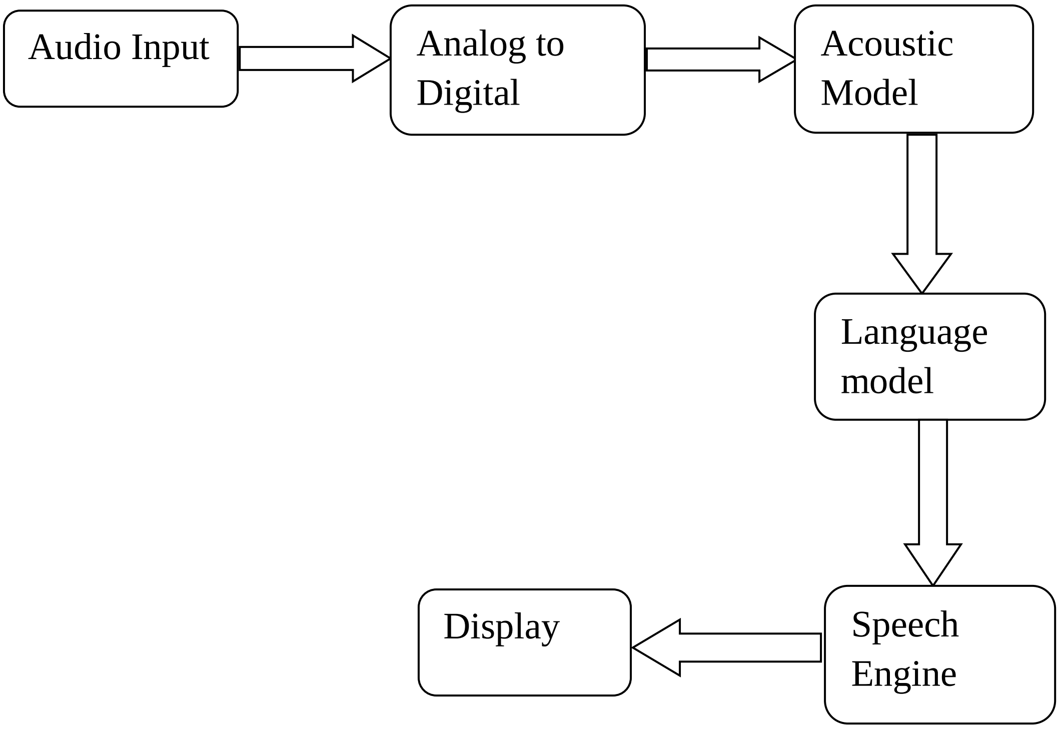
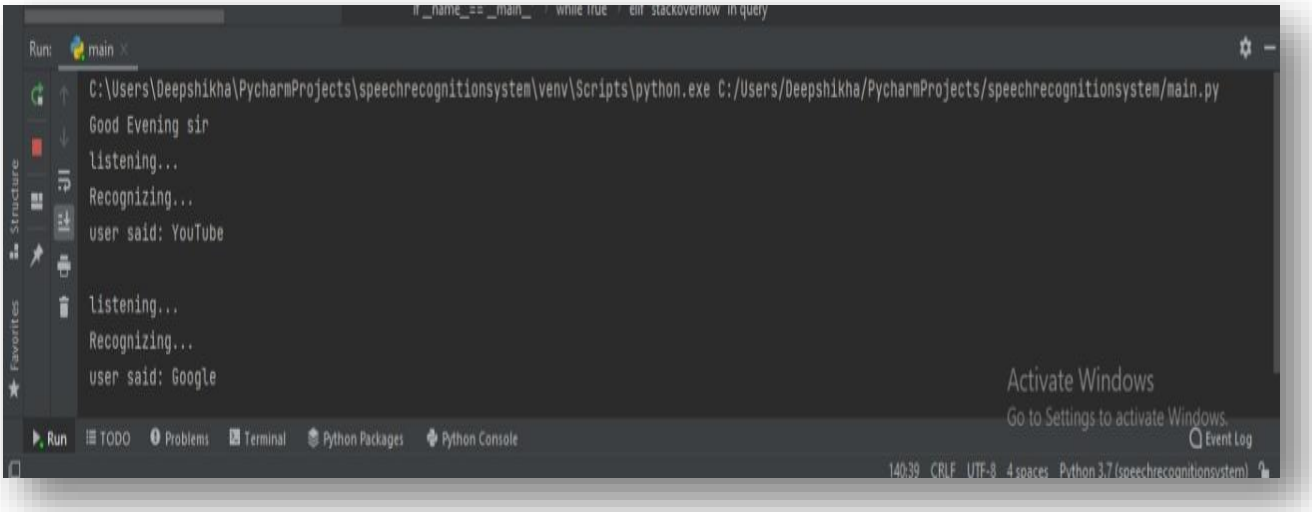


Fig. 1 Speech Recognition Process

INPUT

Here when we speak something like Google (Fig.2) , then firstly our speech recognition system first listen it, and print listening, then recognise it, and print recognizing, and then it print Google as a user said and then execute it. Next figure (Fig.3) gives us output after execution.


if _name_ == _main_ › while True › elif stackoverflow' in query
Run: main
Structure
Favorites
Run
TODO
Problems
Terminal
Python Packages
Python Console
Event Log
Activate Windows
Go to Settings to activate Windows.
140:39 CRLF UTF-8 4 spaces Python 3.7 (speechrecognitionsystem)

```
C:\Users\Deepshikha\PycharmProjects\speechrecognitionsystem\venv\Scripts\python.exe C:/Users/Deepshikha/PycharmProjects/speechrecognitionsystem/main.py
Good Evening sir
listening...
Recognizing...
user said: YouTube

listening...
Recognizing...
user said: Google
```

Fig. 2 Input Google to Speech recognition system

OUTPUT

After Recognizing Google, the model software execute and open Google as we shown in fig. .

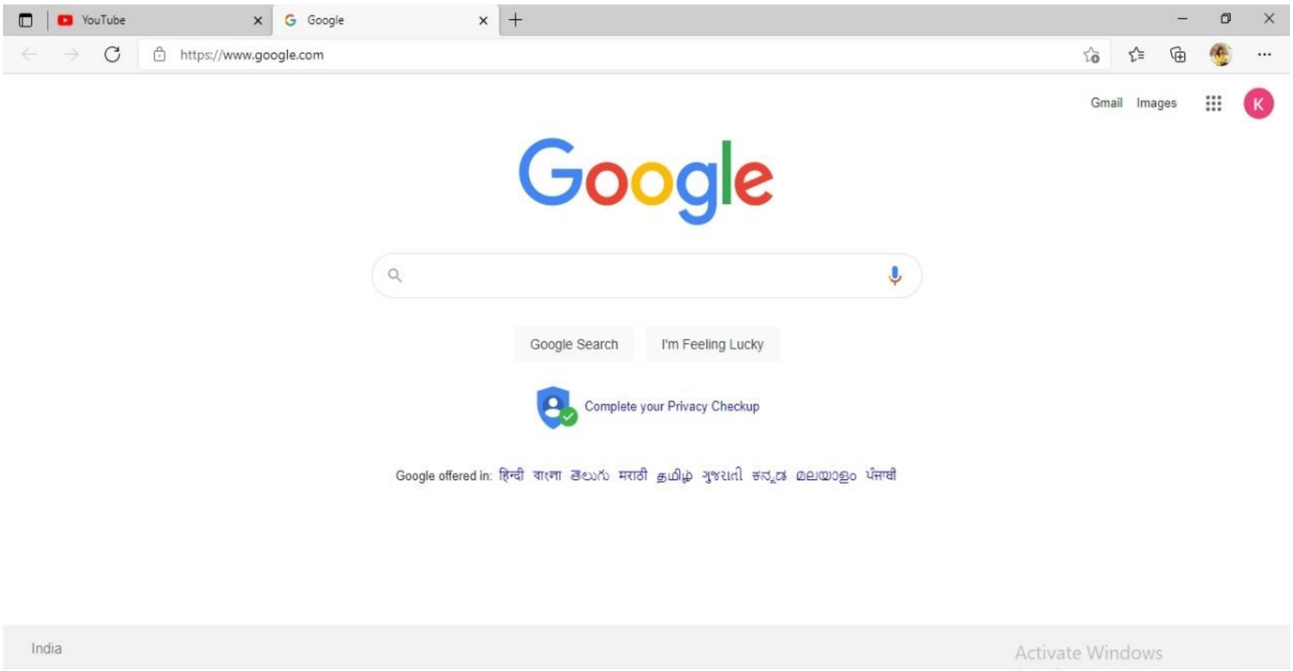
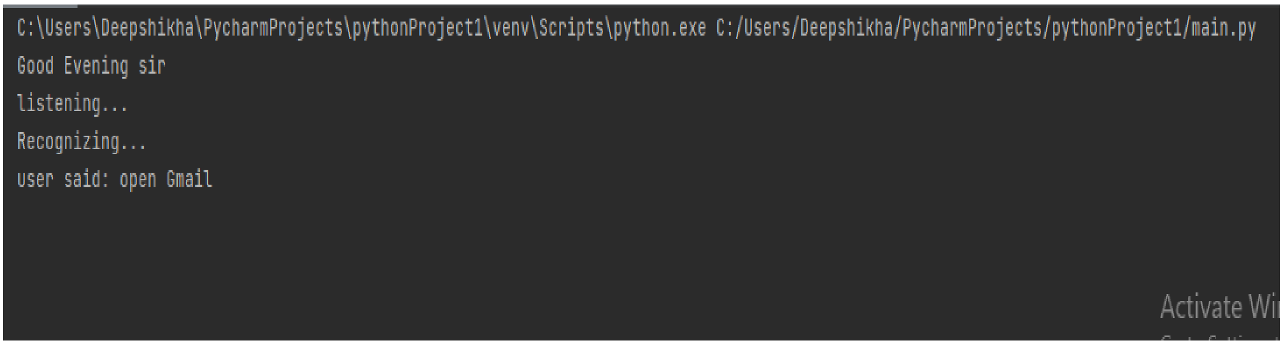

YouTube
Google
https://www.google.com
Gmail
Images
K
Google
Google Search
I'm Feeling Lucky
Complete your Privacy Checkup
Google offered in: हिन्दी বাংলা తెలుగు मराठी தமிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ
India
Activate Windows

Fig. 3 Output of Speech Recognition System

INPUT

Here when we speak something like Gmail (Fig.) , then firstly our speech recognition system first listen it, and print listening, then recognise it, and print recognizing, and then it print open Gmail as a user said and then execute it. Next figure (Fig.) gives us output after execution.

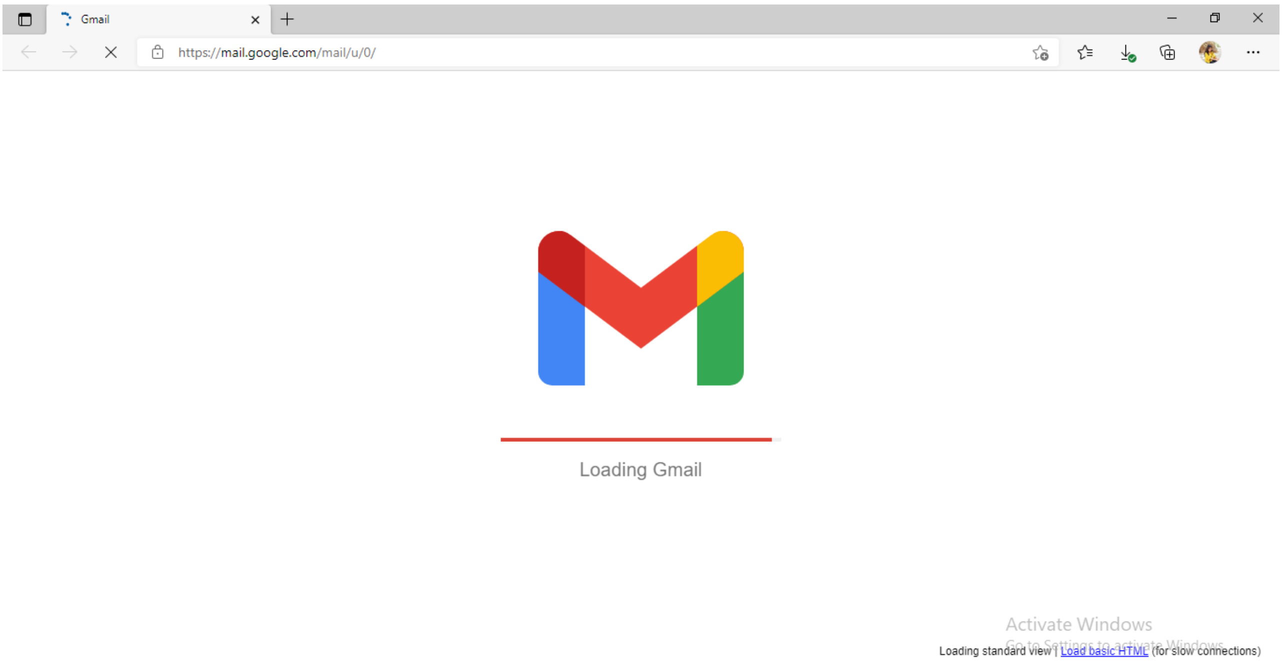


```
C:\Users\Deepshikha\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/Deepshikha/PycharmProjects/pythonProject1/main.py
Good Evening sir
listening...
Recognizing...
user said: open Gmail
```

Fig.4 Input given to Open Gmail

OUTPUT

After Recognizing Gmail, the model software execute and open Gmail as we shown in fig. .



Gmail
https://mail.google.com/mail/u/0/
Activate Windows
Go to Settings to activate Windows.

Loading Gmail

Loading standard view | Load basic HTML (for slow connections)

Fig 5. Gmail is open as output

4.2 FLOW CHART

Flow chart is nothing but a graphical representation of algorithm or work flow or process in any system. Basically it shows the step by step process applied to the system to solving task.

Symbol used in flow chart

Name	Symbol	Description
Flow Line	→	It represent the direction of flow
Process	▭	It shows what operation performed.
Input / Output Symbol	▱	It is used to show the received data and displaying the process data
Decision Box	◇	It is used when decision is necessary.
Start / End Symbol	⬭	It is used to start and end the work flow.

Fig.6 Flowchart symbols

Flow chart diagram

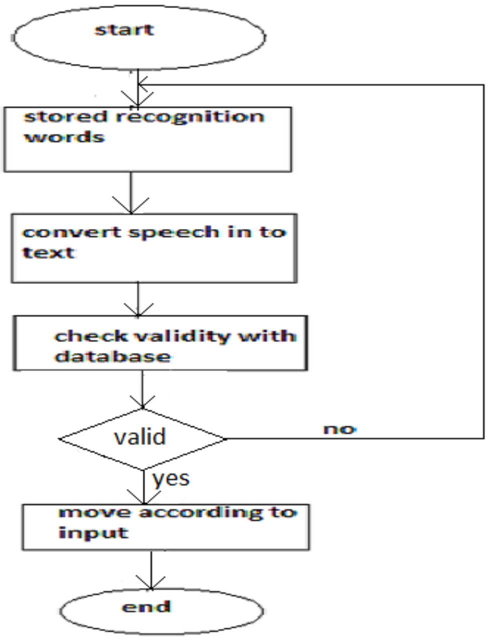


Fig. 7 Diagram of Flow Chart

4.3 USECASE DIAGRAM

Use-Case Diagramcaptures the system's functionality and requirements by using actors and use cases. Use Cases model the services, tasks, function that a system needs to perform. Use cases represent high-level functionalities and how a user will handle the system. Use-cases are the core concepts of Unified Modeling language modeling.

Use-case Diagram notations

Use-case: Use cases are used to represent high-level functionalities and how the user will handle the system. A use case represents a distinct functionality of a system, a component, a package, or a class. It is denoted by an oval shape with the name of a use case written inside the oval shape. The notation of a use case in UML is given below.

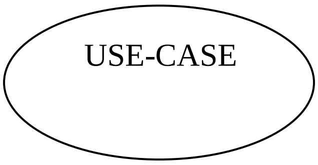


Fig. 8 Use case

Actor: It is used inside use case diagrams. The actor is an entity that interacts with the system. A user is the best example of an actor. An actor is an entity that initiates the use case from outside the scope of a use case. It can be any element that can trigger an interaction with the use case. One actor can be associated with multiple use cases in the system. The actor notation in UML is given below

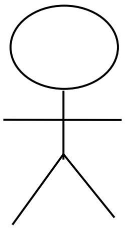


Fig.9 Actor - name

Use case diagram

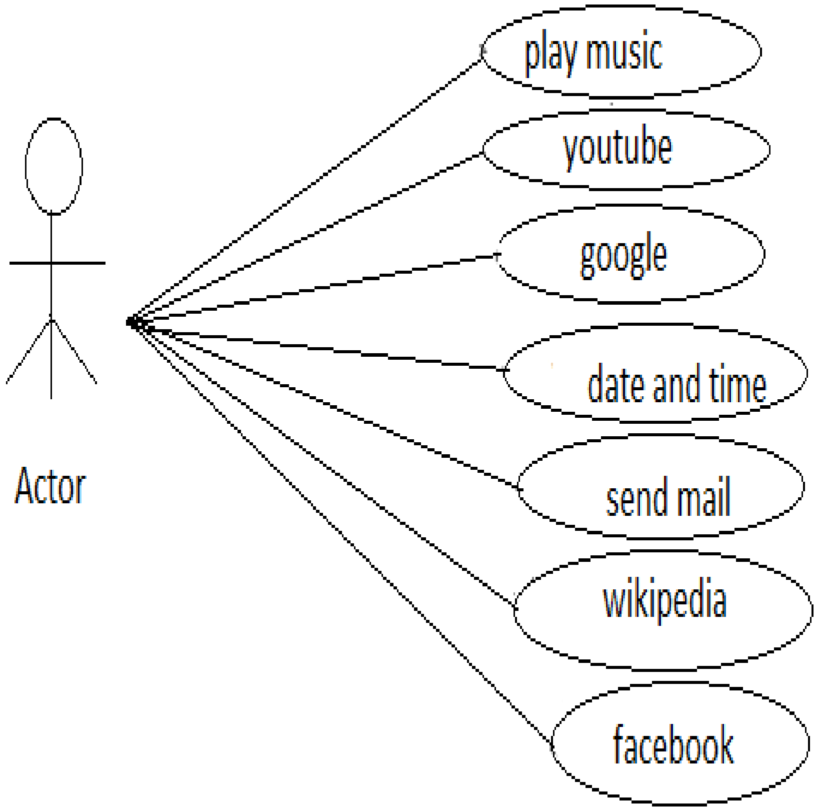


Fig.10 Use Case Diagram

In this project there is only one user. The user queries command to the system. System then interprets it and fetches answer. The user can command the system to open play music, youtube, google, date and tie, send mail, wikipedia, facebook, etc.

4.4 SEQUENCE DIAGRAM

4.4.1 Sequence diagram for Query-Response

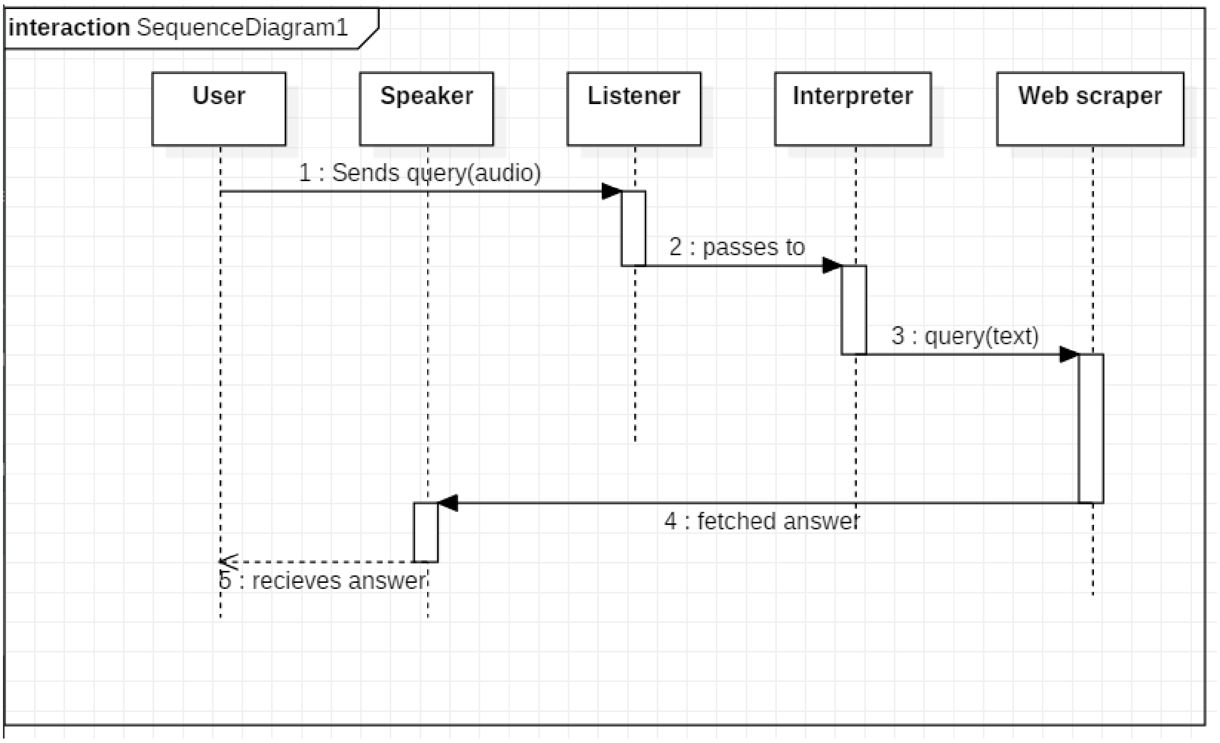


Fig. 11 Sequence Diagram for Query Response

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.

4.4.2 Sequence diagram for Task Execution

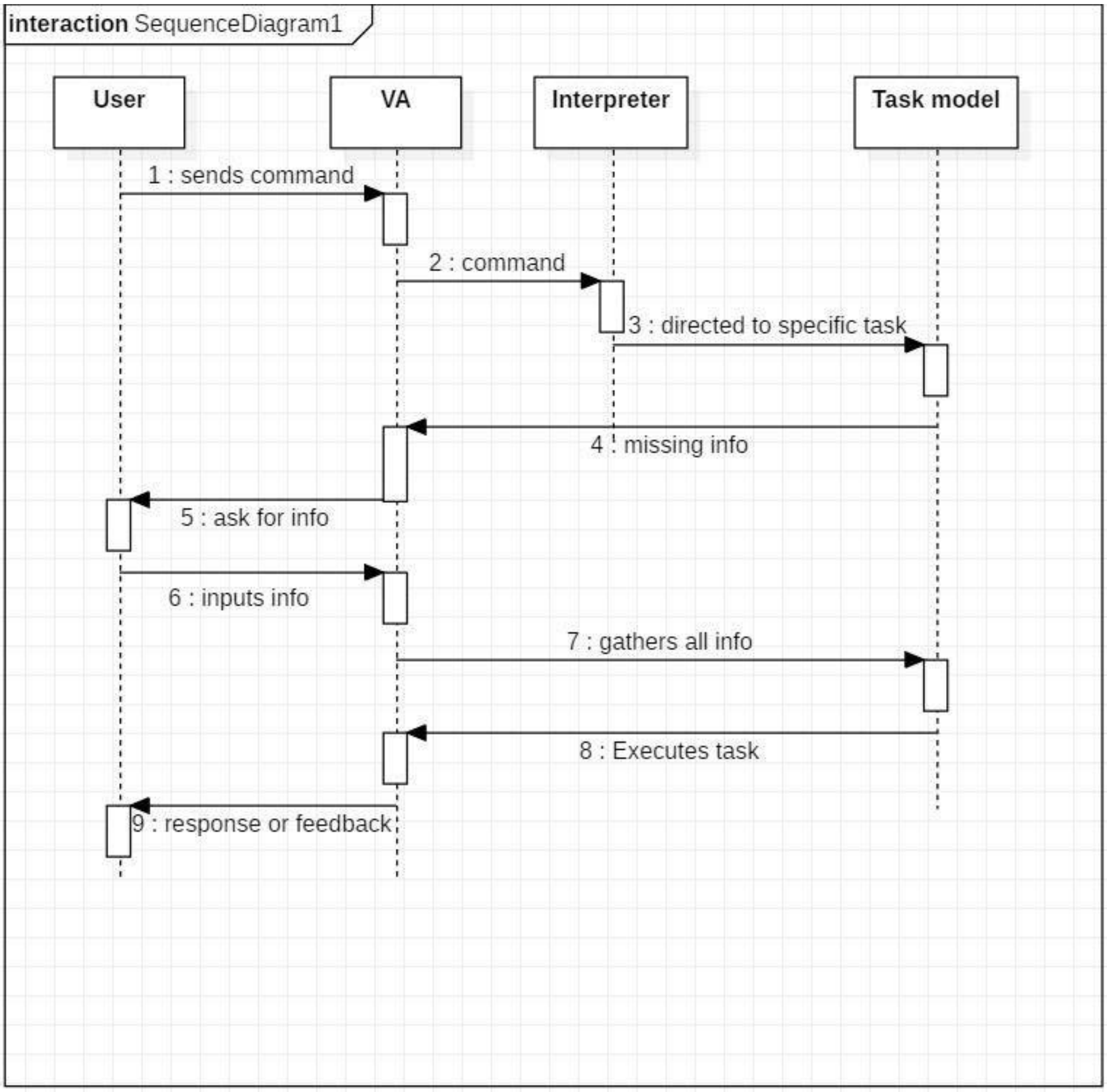


Fig.12 Sequence diagram for Task Execution

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.

4.4 DATA FLOW DIAGRAM

4.5.1 DFD Level 0 (Context Level Diagram)

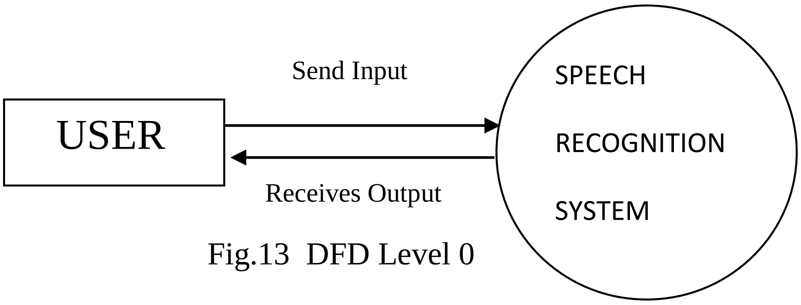


Fig.13 DFD Level 0

4.5.2 DFD Level 1

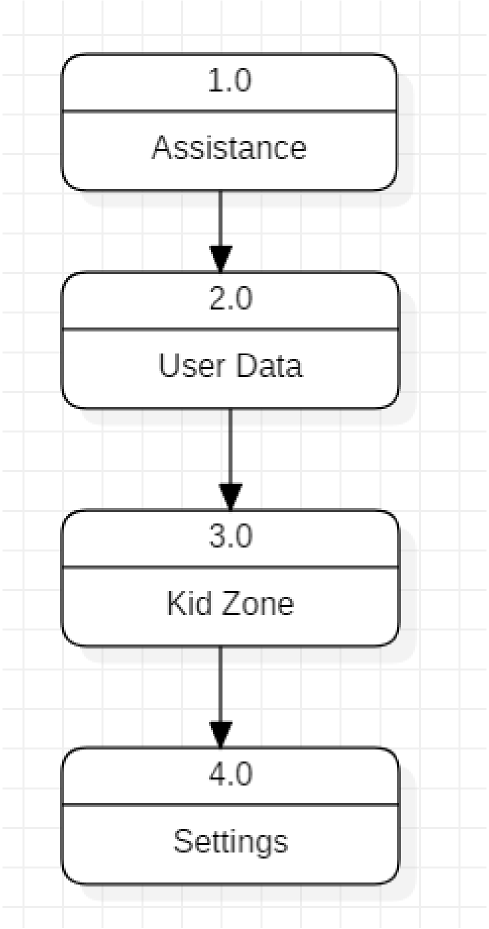


Fig.14 DFD Level 1

4.5.3 DFD Level 2

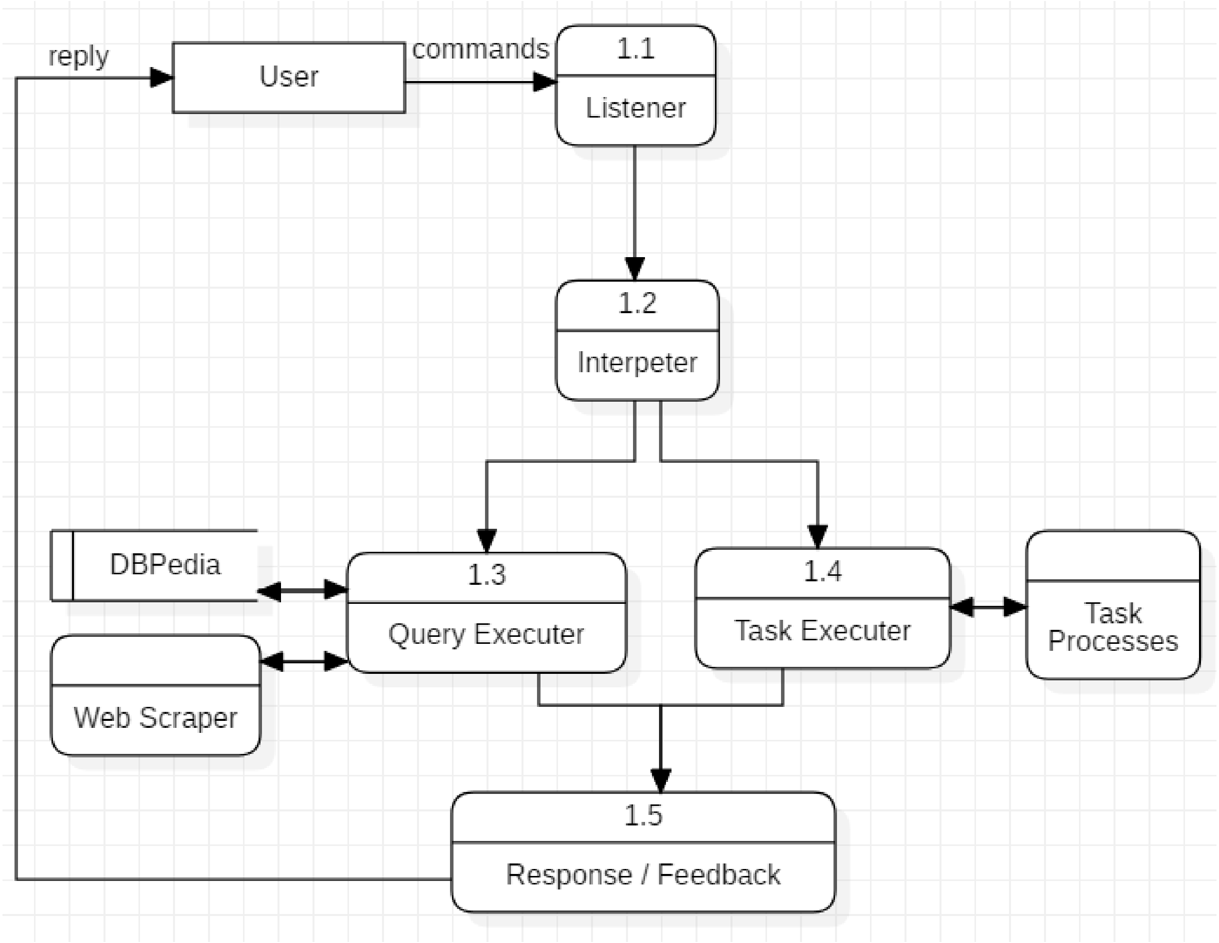


Fig.15 DFD Level 2

4.6 E-R DIAGRAM

What is ER Diagram?

ENTITY-RELATIONSHIP DIAGRAM (ERD**)** displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the model unique.

Components of the E-R Diagram

This model is based on three basic concepts

- Entities
- Attributes
- Relationships

E-R Diagram Notations

- Rectangles: This symbol represent entity types
- Ellipses : Symbol represent attributes
- Diamonds: This symbol represents relationship types
- Lines: It links attributes to entity types and entity types with other relationship types
- Primary key: attributes are underlined
- Double Ellipses: Represent multi-valued attributes

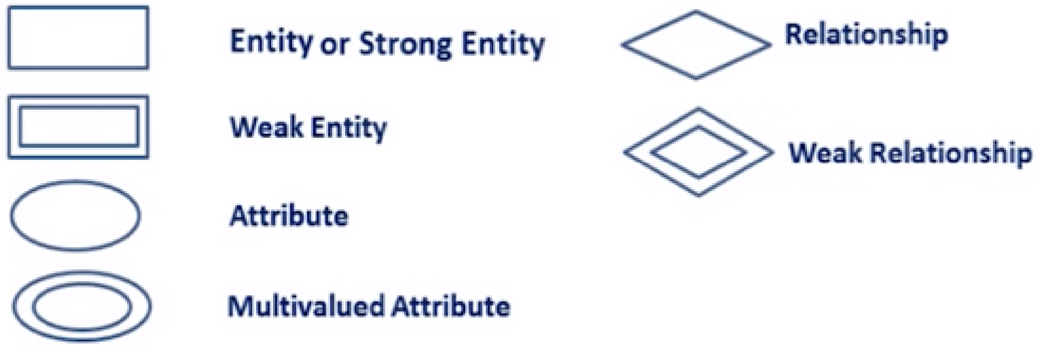


Fig.16 E-R Diagram Notations

E-R Diagram

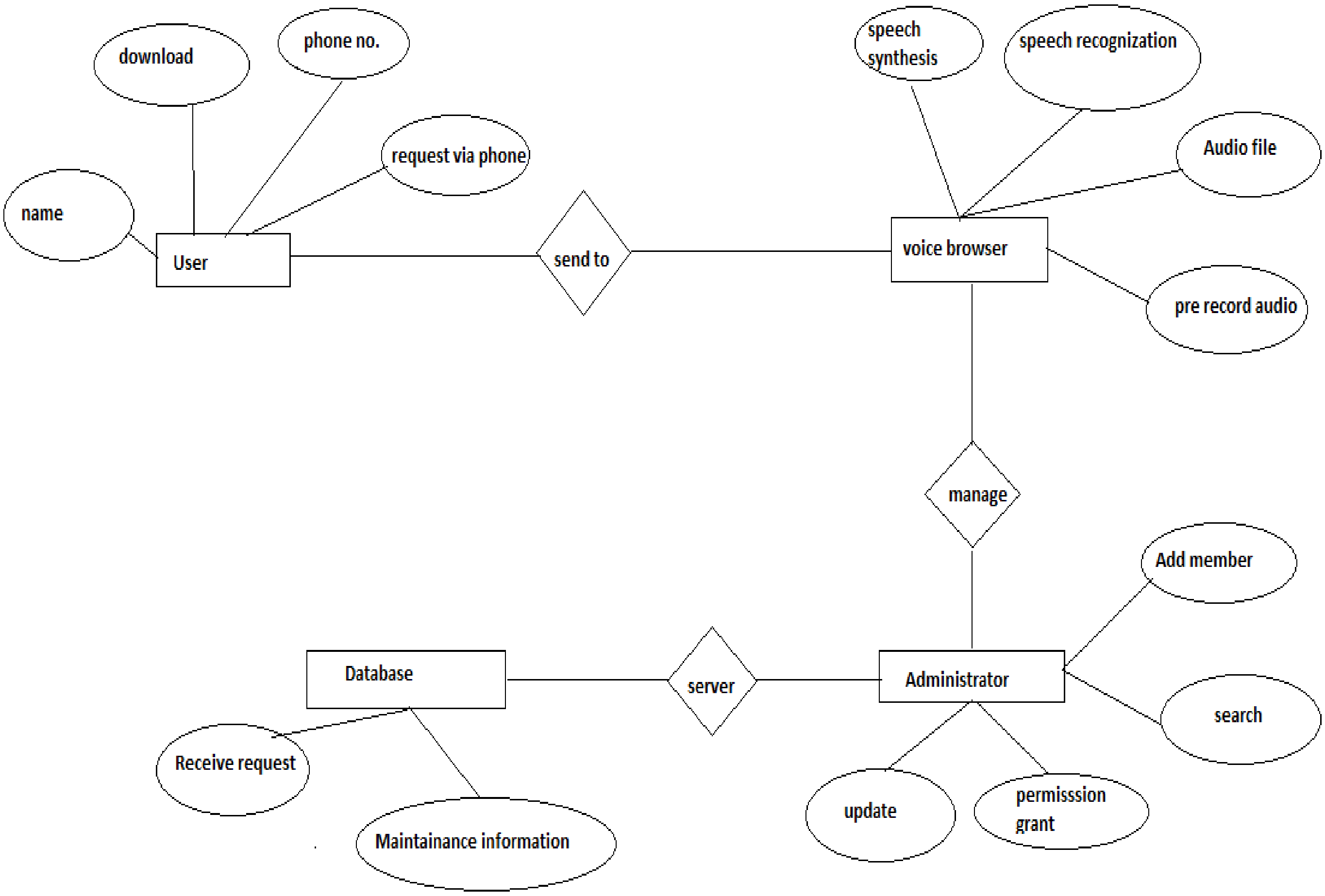


Fig 17. E-R Diagram

The above diagram shows entities and their relationship for a virtual assistant system. We have a user of a system who can have their keys and values. It can be used to store any information about the user. Say, for key "name" value can be "Jim". For some keys user might like to keep secure. There he can enable lock and set a password (voice clip).

Single user can ask multiple questions. Each question will be given ID to get recognized along with the query and its corresponding answer. User can also be having n number of tasks. These should have their own unique id and status i.e. their current state. A task should also have a priority value and its category whether it is a parent task or child task of an older task.

5. CODING

```
import pyttsx3
import speech_recognition as sr
import datetime
import wikipedia
import webbrowser
import os
import smtplib
import random
import sys
import wolframalpha
import psutil
import winshell
import pyjokes
import requests
import pyautogui as pg
import timeit


client = wolframalpha.Client("PXHKTR-5YEP43V3LJ ")

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
# print(voices[1].id)
engine.setProperty('voices', voices[1].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 sir")
        print("Good Morning sir")

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

    else:
        speak("Good Evening sir")
        print("Good Evening sir")
    speak("I am Your Virtual Assistant. Sir.  how may I help you")



def takeCommand():
    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("Try once again please..")
        return " "
    return query

def SR_test(recognizer, microphone, audio):
    command = recognizer.recognize_google(audio)
    print(command)

t = timeit.Timer(lambda: SR_test(recognizer, microphone, audio))
print(t.timeit(0))


def secs2hours(secs):
    mm, ss = divmod(secs, 60)
    hh, mm = divmod(mm, 60)
    return "%dhour, %02d minute, %02s seconds" % (hh, mm, ss)


def wake_word(speak):
    wake_word = ["Hi Buddy", "Hey Ziva", "Hello Bro"]
    speak = speak.lower()
    for i in wake_word:
        if i in speak:
            return True
        return False


def screenshot():
    image = pg.screenshot()
    speak("screen shot taken")
    speak("what do you want to save it as?")
    filename = query()
    image.save(filename + ".png")
    speak("do you want me to show it")
    ans = query()
    if "yes" in ans:
        os.startfile(filename + ".png")
    else:
        speak("never mind")


def sendEmail(to, content):
    server = smtplib.SMTP("smtp.gmail.com", 587)
    server.ehlo()
    server.starttls()
    server.login("sdeepshikha103@gmail.com", "Deepshikha@123")
    server.sendmail("sdeepshikha103@gmail.com", to, content)
    server.close()


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

        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 gmail" in query:
            speak('okay..')
            webbrowser.open("https://gmail.com")


        elif "what's up" in query or "how are you" in query:
            stMsgs = ["I am 100% fit sir!", "I am fine!", "Nice!", "Bingo Great Sir lets rock it!"]
            speak(random.choice(stMsgs))

        elif "youtube" in query:
            speak('okay..')
            webbrowser.open("https://youtube.com")

        elif "google" in query:
            speak('okay..')
            webbrowser.open("https://google.com")

        elif "github" in query:
            speak('okay..')
            webbrowser.open("https://github.com")

        elif "Facebook" in query:
            speak('okay..')
            webbrowser.open("https://facebook.com")
```

```
elif "stack overflow" in query:
    speak("okay..")
    webbrowser.open("https://stackoverflow.com")

elif "news" in query:
    speak("okay..")
    news= webbrowser.open_new_tab("https://timesofindia.indiatimes.com/home/headlines")
    speak("here are some headlines from times of india, happy reading")

elif "hotstar" in query:
    speak("okay..")
    webbrowser.open("https://www.hotstar.com/in")

elif "instagram" in query:
    speak("okay..")
    webbrowser.open("https://www.instagram.com/")

elif "flipkart" in query:
    speak("okay..")
    webbrowser.open("https://www.flipkart.com/")

elif "amazon" in query:
    speak("okay..")
    webbrowser.open("https://www.amazon.in/")

elif "play music" in query:
    music_dir = "D:\Deepshikha\Music"
    songs = os.listdir(music_dir)
    print(songs)
    os.startfile(os.path.join(music_dir, songs[1]))


elif "video" in query:
    video_dir = "D:\Deepshikha\Videos"
    video = os.listdir(video_dir)
    os.startfile(os.path.join(video_dir, video[1]))


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

elif " find location" in query:
    location = speak("What is the location ?")
    url = "https://www.google.nl/maps/place" + place + "/&amp;"
    webbrowser.get().open(url)
    speak(f'Here is the location of' + place)
```

```
elif "weather" in query:
    api_key = "3c1b87bfa7f9b6106d313cd7e64ee9f6"
    base_url = "http://api.openweathermap.org/data/2.5/weather?"
    speak("city name")
    print("city name : ")
    city_name = takeCommand()
    complete_url = base_url + "appid=" + api_key + "&q=" + city_name
    response = requests.get(complete_url)

    x = response.json()

    if x["cod"]!="404":
        y = x["main"]
        current_temperature = y["temp"]
        current_pressure = y["pressure"]
        current_humidity = y["humidity"]
        z = x["weather"]
        weather_description = z[0]["description"]

        print(" Temperature (in kelvin unit) = " +
          str(current_temperature) +
          "\n atmospheric pressure (in hPa unit) = " +
          str(current_pressure) +
          "\n humidity (in percentage) = " +
          str(current_humidity) +
          "\n description = " +
          str(weather_description))

    else:
          print(" City Not Found ")

elif 'empty recycle bin' in query:
    winshell.recycle_bin().empty(confirm=False, show_progress=False, sound=True)
    speak("Recycle Bin Recycled")

elif "thanks and quit" in query or "stop" in query or "take rest" in query:
    speak("Okay.. Bye Sir.. dhanyawad ..")
    sys.exit()

elif "shutddown" in query:
    speak('okay.. Sir! shutting down your pc')
    os.system("shutdown/s")

elif "restart" in query:
    speak("Okay.. Sir restarting your pc")
    os.system("shutdown/r")

elif "hibernate" in query:
    speak("Hibernating")
    os.system("shutdown/h")
```

```
elif "battery percent" in query:
    battery = psutil.sensors_battery()
    plugged = battery.power_plugged
    percent = str(battery.percent)
    if plugged == False:
        plugged == "not plugged in"
    else:
        plugged = "plugged in"
        print(percent + "percentage of battery", plugged)
        speak(percent)

elif " search " in query:
    speak("what do you want to search")
    url = 'https://google.com/search?q=' + search
    webbrowser.get().open(url)
    speak("here is what i found for you" + search)

elif "joke" in query:
    speak(pyjokes.get_joke())

elif "mail" in query:
    try:
        speak("what should I say")
        content = takeCommand()
        to = "ashishyahoo556@gmail.com"
        sendEmail(to, content)
        speak("Email has been sent")
    except Exception as e:
        print(e)
        speak("Sorry Sir i am not able to send this email")


elif "how are you" in query:
    speak("I am fine , Thank you")
    speak("How are you sir")

elif "fine" in query or "good" in query:
    speak("It is good to know that you are fine")


elif "who made you" in query:
    speak("i was built by Information Technology Department")

else:
    query = query
    speak("searching..")
    try:
        res = client.query(query)
        results = next(res.results).text
        speak("Wolfram-Alpha says...")
        speak("okay  got it..")
        print(results)
        speak(results)
    except Exception as e:
        print(e)
        speak("Sorry Sir I am not able to get this")

speak("Next command sir")
```

6. TESTING

- **Test Case 1**

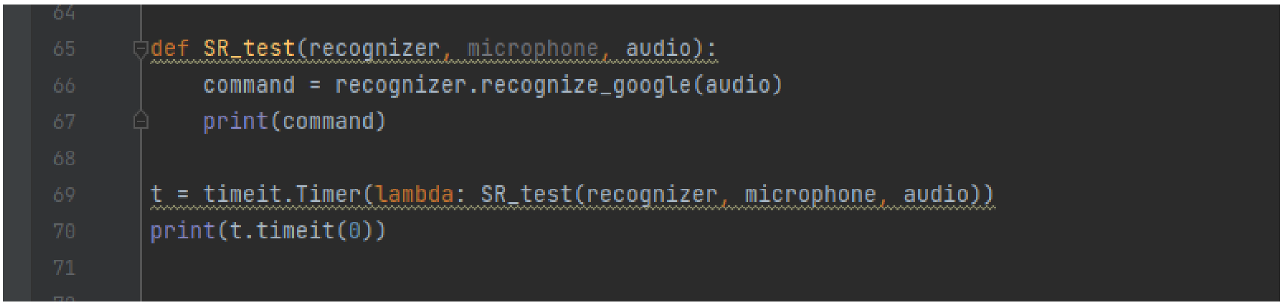
Test Title: Response Time

Test Id: T1

Test Priority: High

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

Test Code:

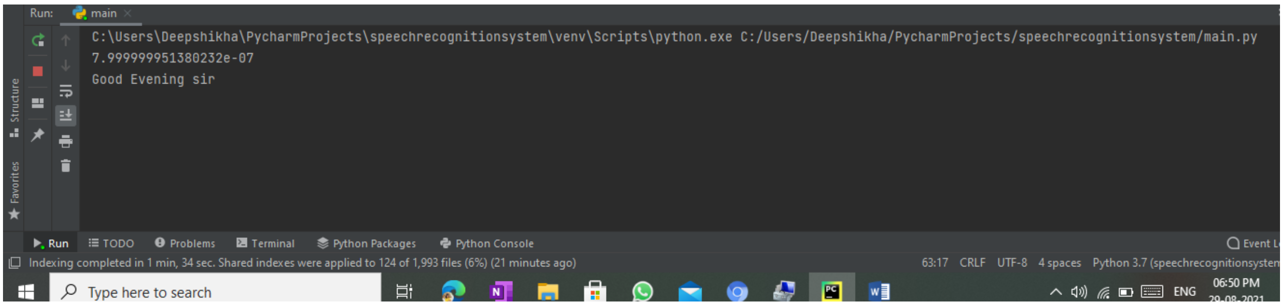


```python
def SR_test(recognizer, microphone, audio):
    command = recognizer.recognize_google(audio)
    print(command)

t = timeit.Timer(lambda: SR_test(recognizer, microphone, audio))
print(t.timeit(0))
```

Fig 18. Test Code

Test Output:



```
C:\Users\Deepshikha\PycharmProjects\speechrecognitionsystem\venv\Scripts\python.exe C:/Users/Deepshikha/PycharmProjects/speechrecognitionsystem/main.py
7.999999951380232e-07
Good Evening sir
```

```
Indexing completed in 1 min, 34 sec. Shared indexes were applied to 124 of 1,993 files (6%) (21 minutes ago)
63:17   CRLF   UTF-8   4 spaces   Python 3.7 (speechrecognitionsysten
06:50 PM
29-08-2021
```

Fig 19. Test Output

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. User must get instant response of the query made.

- **Test Case 2**

Test Title: Accuracy

Test ID: T2

Test Priority: High (70%-80%)

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

Description:

A Speech Recognition system is mainly used to get precise answers to any question asked. Getting answer in a moment is of no use if the answer is not correct. Accuracy is of utmost importance in a speech recognition system.

- **Test Case**

Test Title: Approximation

Test Id: T3

Test priority: Moderate

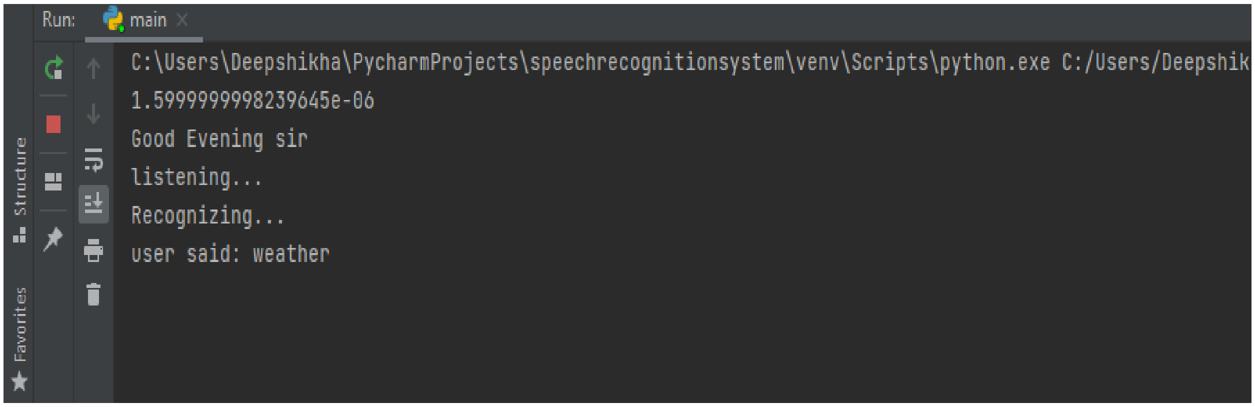
Test Objective: To check approximate answers about calculations.

Description:

There are times when mathematical calculation requires approximate value. For example, if someone asks for value of PI the system must respond with approximate value and not the accurate value. Getting exact value in such cases is undesirable.

7. Result

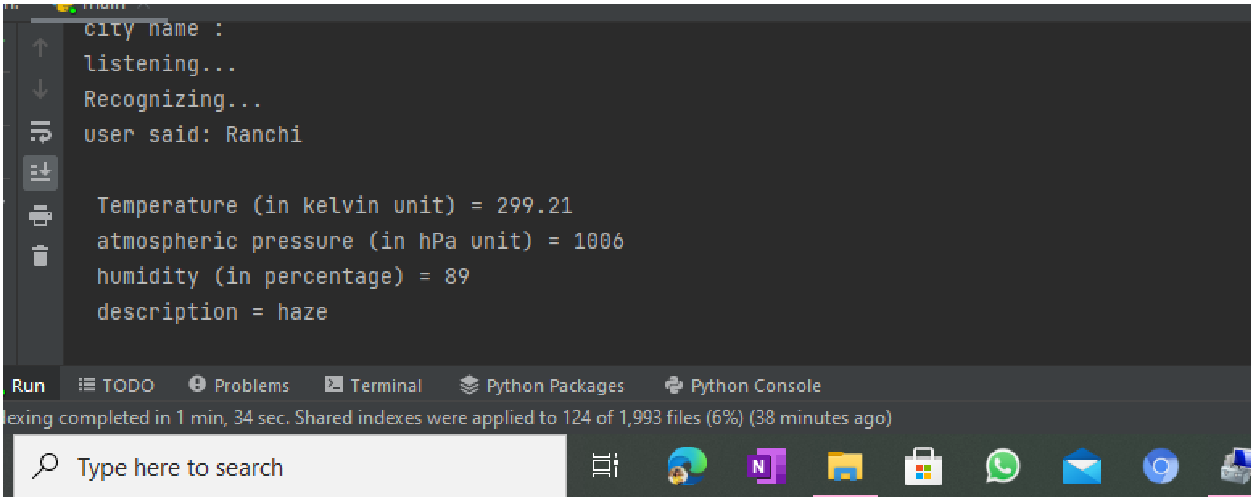
Input:



```
C:\Users\Deepshikha\PycharmProjects\speechrecognitionsystem\venv\Scripts\python.exe C:/Users/Deepshik
1.5999999998239645e-06
Good Evening sir
listening...
Recognizing...
user said: weather
```

Fig 20. Input Result

Output:



```
city name :
listening...
Recognizing...
user said: Ranchi

 Temperature (in kelvin unit) = 299.21
 atmospheric pressure (in hPa unit) = 1006
 humidity (in percentage) = 89
 description = haze
```

lexing completed in 1 min, 34 sec. Shared indexes were applied to 124 of 1,993 files (6%) (38 minutes ago)

Fig 21. Output Result

8. Conclusion and future scope

Conclusion

Researchers, working on the very promising and challenging field of automatic speech recognition, are collectively heading towards the ultimate goal i.e. Natural Conversation between Human beings and machines, are applying the knowledge from areas of Neural Networks, Psychoacoustics, Linguistics, Speech Perception, Artificial Intelligence, Acoustic-Phonetics etc.. The challenges to the recognition performance of ASR are being provided concrete solutions so that the gap between recognition capability of machine and that of a human being can be reduced to maximum extent. An attempt has been made through this paper to give a comprehensive survey and growth of automatic speech recognition over the last six decades through the never ending efforts of researchers in countries like China, Russian, Portuguese, Spain, Saudi Arab, Vietnam, Japan, UK, Sri-Lanka, Philippines, Algeria and India.

Future Scope

Speech Recognition System 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 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 the speech recognition system is missing. Users simply cannot see or touch a voice interface

9. References

- Websites referred
  - www.stackoverflow.com
  - www.pythonprogramming.net
  - www.codecademy.com
  - www.tutorialspoint.com
  - www.google.co.in
  - www.geekof geeks.com

- Books referred
  - Python Programming - Kiran Gurbani
  - Learning Python - Mark Lutz

- YouTube Channels referred
  - CS Dojo
  - edureka!
  - Code with Harry

- Documents referred
  - Designing Personal Assistant Software for Task Management using SemanticWeb Technologies and Knowledge Databases
    - Purushotham Botla
  - Python code for Artificial Intelligence: Foundations of Computational Agents
    - David L. Poole and Alan K. Mackworth.