



DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

**A
MINI PROJECT REPORT**

ON

**“VOICE BASED EMAIL SYSTEM FOR VISUALLY
CHALLENGED USING PYTHON”**

Submitted in the partial fulfillment of the requirements in the 5th semester of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

BY

**SOUJANYA S
1NH17IS105**

Under the guidance of

Mrs. Mounica B,
Sr. Assistant Professor,
Dept. of ISE, NHCE

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

NEW HORIZON COLLEGE OF ENGINEERING

(Autonomous College Permanently Affiliated to VTU, Approved by AICTE, Accredited by
NBA & NAAC with 'A' Grade)

Ring Road, Bellandur Post, Near Marathalli,
Bangalore-560103, INDIA



CERTIFICATE

Certified that, the mini project report entitled **“VOICE BASED EMAIL SYSTEM FOR VISUALLY CHALLENGED USING PYTHON”** carried out by SOUJANYA S (1NH17IS105), a bonafied student of New Horizon College of Engineering, Bengaluru, in partial fulfillment of the requirements in the V semester of Bachelor of Engineering in Information Science and Engineering the year 2019-2020. The project report has been approved as it satisfies the academic requirement in respect of mini project work.

Name & Signature of student

(Ms. Soujanya S)

Name & Signature of Guide

(Mrs. Mounica B)

Name & signature of HOD

(Dr. R J Anandhi)

ABSTRACT

In the present world, Internet has developed to be one of the most basic comforts for many people. Every human being is extensively using the internet to acquire knowledge, information as well as to communicate. Internet has become a very important necessity in our lives. One of the features provided to us by the internet is communication that has become so easy due to combination of communication technologies with internet. Communication plays a very important role in a person's professional, social and personal life. In spite of all the advantages provided to us by the internet, it is still difficult or near to impossible for the visually challenged people to access the all the features provided by the internet.

But the progression in the computer based technologies has undone many opportunities for the visually impaired across the globe in an extensive manner. These progressions ensure that the visually impaired will no longer have to depend on anyone for anything. As mentioned earlier communication plays a very important role in a person's professional, social and personal life. Communication through emails is still one of the most formal methods to communicate over the internet. Therefore developing an application that can enable the blind to send and receive emails independently without the help of anyone is crucial.

The application titled "Voice Based E-mail for the Visually Impaired" is a web-based application developed in Python that allows the users to compose a mail and check their inbox through the Interactive Voice Response technology. This application not only prompts the users with voice commands but also takes their voice input, converts it into text using the features like Text To Speech, Speech to Text, Speech Recognition, pyaudio, beautifulsoup4, gTTS and pygame features provided by python. The main advantage of the above proposed system is that the use of keyboard and mouse is completely eliminated; that is, the user will have to respond through voice commands only.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project. I would like to take an opportunity to thank them all.

I thank the management, **Dr. Mohan Manghnani**, Chairman, New Horizon Educational Institutions for providing necessary infrastructure and creating good conducive environment for effective learning.

I also here record constant encouragement support and facilities extended to us by **Dr. Manjunatha**, Principal, New Horizon College of Engineering, Bengaluru.

I extend my sincere gratitude for the constant encouragement support and facilities provided to us by **Dr. R J Anandhi**, Professor and Head of the Department, Department of Information Science and Engineering, New Horizon College of Engineering, Bengaluru.

I sincerely acknowledge the encouragement, timely help and guidance to me by **Mrs. Mounica B**, Sr. Assistant Professor, Department of ISE, NHCE, and Bengaluru, to complete the mini project within the stipulated time.

Finally, a note of thanks to the teaching and non-teaching staff of Information Science and Engineering Department for their cooperation extended to us and our friends, who helped me directly or indirectly in the successful completion of this mini project.

SOUJANYA S
(1NH17IS105)

TABLE OF CONTENTS

Abstract	i
Acknowledgement	ii
Table of contents	iii
List of figures and tables	iv
Chapters	
Chapter 01: Introduction	01
1.1 Purpose of study	01
1.2 Problem statement	02
1.3 Motivation	02
1.4 Methodology	03
Chapter 02: System Requirements and Specification	04
2.1 Hardware Specifications	04
2.2 Software specifications	04
2.3 About the Language	04
Chapter 03: System Design	08
3.1Architecture	08
3.2 Algorithm	09
3.3 Flowchart	10
3.4 Code	11
Chapter 04: Results and Discussion	21
4.1. Summary of result obtained	21
4.2 Output	21
Chapter 05: Conclusion	23
References	24

LIST OF FIGURES AND TABLES

Fig 3.1.1	SYSTEM ARCHITECTURE	08
Fig 3.1.2	FLOWCHART OF THE PROGRAM	10
Fig 4.2 .1	OUTPUT WHEN THE USER OPTS OPTION 1	21
Fig 4.2.2	THE MAIL HAS BEEN SENT SUCCESSULLY	22
Fig 4.2.3	OUTPUT WHEN THE USER OPTS FOR OPTION 2	22

Chapter 01

INTRODUCTION

The mini project titled “Voice Based E-mail for the Visually Challenged” is an application developed in Python3.7.7. This application is developed to aid the visually challenged to use their Gmail accounts without the help of anyone. This can have a huge impact on the visually impaired as communication can help people, independent of their physical appearance, that is, physically abled or normal people, to build their social, professional, personal lives. We all realize that the internet can play a very important role in the present world of communication. Currently the world is functioning on the internet. Everything in the today’s world requires the use of internet. The life of many people has become so easy because of the internet. Communication is one of the many features that have been highly transformed by the Internet. The introduction of electronic mails or conveniently called as e-mail has proved to be a boon in the field of communication. The business or formal communication, even now, uses the concept of e-mails extensively. Nevertheless, there are specially abled people around us who are not gifted with the ability to see, that is, there are a few visually challenged who cannot see the things around them. A survey conducted recently determines that there are over 240 million visually challenged people everywhere in the globe, which means; it is a very difficult task for about 240 million people to use Internet or E-mail.

1.1. PURPOSE OF STUDY

The above proposed system is a python based application that is specifically designed for the visually challenged people. As the name suggests, this application provides a voice based email service that enables the user to send mails and check their inbox without any help, through their g-mail accounts. The introduction of electronic mails or conveniently called as e-mail has proved to be a milestone in the field of communication. The business or formal communication, even now, uses the concept of e-mails extensively. Nevertheless, there are physically challenged people around us who are not gifted with the ability to see, that is, there are a few visually challenged who

cannot see the things around them. The main aim of the above proposed system is to develop an email system that will enable even a naïve or visually challenged person to exploit the communication services provided by the internet. This system eliminates the use of keyboard and mouse and will work only on the voice commands

1.2. PROBLEM STATEMENT

To design and implement a Voice Based E-mail System for the visually challenged using Python.

1.3. MOTIVATION

Around 4.1 billion Gmail accounts have been created until 2014 and it is predicted that around 5.2 billion accounts will be created by 2019. This clearly indicates that the emails still remain to be the extensively used method of communication on internet. Such widely used mailing services cannot be used by the visually impaired. The reason can be that they fail to provide any facilities so that an individual in the front can listen to the content to the screen. As it is difficult for them to visualize the contents already existing on the screen, they cannot determine where to click or what to do in order to accomplish the necessary operations. For a blind person using a computer is not that easy as it is for an ordinary user though it claims to be user friendly, though there are several screen readers available. The available screen readers will read out or dictate everything present on the screen and to achieve required activities the person will have to use the keyboard shortcuts as the mouse cursor location cannot be determined by the screen readers. This can either means that the user cannot use the mouse pointer as it is absolutely difficult if the pointer location is not traced or that the user must be well versed with the keyboard and know where every key is located. To summarize, in the existing system the users' needs to use the computer on visual perceptions and proves to be troublesome for the blind. The above proposed system is established on a entirely different idea and is different from the existing mail systems. A web system or any application or services is said to be flawless or completely accessible only if it can be exploited efficiently by all sorts of people independent of a person being able or disable. The existing systems fail to provide this accessibility. Therefore the above proposed

system proves to be different from the existing system. Unlike the existing system that accentuates more on user friendliness of ordinary users, this system emphasizes more on user friendliness of all sorts of people comprising ordinary people, visually compromised people as well as illiterate people. The whole system is based on IVR- interactive voice response. While using the above proposed system the system will prompt the user to accomplish certain processes to avail respective services. One of the key benefits of this system is that usage of the keyboard and mouse is completely eliminated and all the operations will be based on voice commands. Therefore the above proposed system is perfectly accessible to all sorts of users as it is based on speech inputs and therefore eliminating the need of remembering keyboard shortcuts.

1.4. METHODOLOGY

The above proposed system allows the users to compose a mail and check their inbox through an Interactive Voice Response technology. This application not only prompts the users with voice commands but also takes their voice input, converts it into text using the features like Text To Speech, Speech to Text. One of the main advantages of the above proposed system is that the use of keyboard and mouse is completely eliminated; that is, the user will have to respond through voice commands only.

INTERACTIVE VOICE RESPONSE

Interactive voice response abbreviated as IVR is a technology that lets a system to interact with the users via voice commands. In the field of telecommunications, IVR permits the clients to interact with the corporation's host system through telephone keypad or speech recognition. IVR systems will be able to answer back with pre-recorded and dynamically produced audio to direct the consumers on the way to continue. IVR systems installed in the system are sized such that they will be able to handle huge call capacities and is used for outbound calling, as the IVR systems prove to be quicker and more efficient than many of the predictive dialer systems. These systems are used for mobile purchases, banking services, retail orders, travel data and climatic situations. The main purpose of an IVR is to acquire the input from the user using voice commands, process the input and give back the results.

Chapter 02

SYSTEM REQUIREMENTS

2.1. HARDWARE AND SOFTWARE REQUIREMENTS

- Hardware System Configuration:
 1. Processor - Intel Core i5
 2. Speed - 1.8 GHz
 3. RAM - 256 MB (min)
 4. Hard Disk - 10 GB

- Software System Configuration:
 1. Operating System - Windows
 2. Programming Language - Python
 3. Compiler - PyCharm

2.2. ABOUT THE LANGUAGE

Python is a high-level programming language. It is a general purpose language, which is interactive and is an object-oriented scripting language that is extensively used. This language was developed by Guido van Rossum in 1985. Python was primarily developed to improve the readability of the code. So as to achieve this, the program allows the users to express various conceptions in lesser lines of code. Python enables the user to function quickly and efficiently. Some of the key features of python include:

- It not only supports object oriented programming but also structural and functional programming.
- It includes fewer keywords, and has very simple structure. The main advantage of python is that it has a well-defined syntax.
- It is very easy to maintain.
- Python is highly portable.

- Python has special feature of automatic garbage collection.
- It can be easily integrated with other programming languages like C, C++ and many more.

SPEECH RECOGNITION

Speech recognition is a special feature that is availed by python language. It is an interdisciplinary field of computational linguistics which develops methods and technologies that permits the recognition as well as the translation of vocal language into text by systems. Speech Recognition is a significant feature in numerous applications used such as home automation, artificial intelligence and many more. It can be called as automatic speech recognition abbreviated to ASR, computer speech recognition, or only speech to text abbreviated to (STT). Speech recognition applications include voice user interfaces such as voice dialing, call routing, domestic appliance control, search, simple data entry, preparation of structured documents, speech-to-text processing, and aircraft. Libraries that can be used for performing speech recognition, by the provision provided for numerous engines and APIs, both online and offline are Microsoft Bing Voice Recognition, CMU Sphinx that works offline, Wit.ai, Google Speech Recognition, Houndify AP, Google Cloud Speech API, Snowboy Hotword Detection that works offline and IBM Speech to Text. To install speech recognition library we can use the command `pip install SpeechRecognition` in the command prompt. It works by means of algorithms via acoustic and language modeling. Acoustic modeling signifies the connection between linguistic units of speech as well as audio signals whereas language modeling compares sounds with word arrangements to help differentiate between synonyms. Frequently, hidden Markov models are also used to identify sequential patterns in speech to improve correctness in the system. The most common applications of speech recognition in the enterprise comprise call routing, voice dialing, speech-to-text processing and voice search. The performance Speech recognition is represented by factors like accuracy and speed. Accuracy is measured through word error rate. WER mechanisms at the word level and recognize inaccuracies in record, though it fails to recognize in what way the error occurred. Speed is measured through the real-time factor. A range of factors can affect mainframe speech recognition performance that

includes pronunciation of a person, his accent, his pitch, his volume and the background noise. Speech Recognition is the one of the many libraries that is compatible with Python 3.3+, but it requires a few extra installations. For executing the above mentioned project Python v3.6.7 is used. We can install speech recognition and pyaudio is the following steps.

1.>shell-\$ pip install SpeechRecognition.

2.>shell-\$ pip install python3-pyaudio.

The library, speech recognition, works well with existing audio files. The pyaudio package can be used when a microphone should be used to capture the input.

SPEECH TO TEXT- STT

The system gets speech at the execution time via microphone and processes the speech to recognize the spoken text. The recognized text is stored in a file. This is developed on Android platform using Eclipse workbench. The speech-to-text system directly obtains and changes speech to text. It can enhance other greater systems, giving users different choices for data entry. This speech-to-text system can improve system convenience by providing data entry options for disabled like blind, deaf or physically compromised users. Speech recognition system is divided into blocks like feature extraction, acoustic models database which is built based on the training data, dictionary, language model and the speech recognition algorithm. Analog speech signal must first be sampled at time and amplitude axes, or digitized. Samples of the speech signal are analyzed in even intervals. This period is usually 20 ms because the signal in this interval is considered stationary. Speech feature extraction involves the formation of equally spaced discrete vectors of speech characteristics. Feature vectors from training database are used to estimate the parameters of acoustic models. The acoustic model describes properties of the basic elements that can be recognized. The basic element can be a phoneme for continuous speech or word for isolated words recognition. Every word that is spoken will appear on the screen in an accessible format, although one can request a change in the color and font size. As well as every word spoken, the words "new speaker:" will typically appear to denote when the speaker changes. If one sends the STTR the names of people

attending the conference or meeting before the event, they, too, can be programmed into the computer, making it easier for one to recognize who is speaking. Occasional monodegreen errors may be seen in closed-captions when the computer software fails to distinguish where a word break occurs in the syllable stream.

TEXT TO SPEECH-TTS

Speech synthesis or text to speech is the synthetic production of speech. An automatic data handing out system used for this purpose is called as speech synthesizer, and may be enforced in software package and hardware product. A text-to-speech (TTS) system converts language text into speech, alternative systems render symbolic linguistic representations. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly. An intelligible text to speech program permits individual with ocular wreckage or reading disabilities to concentrate to written words on a computing device. Several computer operational systems have enclosed speech synthesizers since the first nineteen nineties years. This method is commonly known as text, standardization, or processing. Front end then assigns spoken transcriptions to every word, and divides and marks the text into speech units, like phrases, clauses, and sentences. Text to speech (TTS) is the use of software to create an audio output in the form of a spoken voice. The program that is used by programs to change text on the page to an audio output of the spoken voice is normally a text to speech engine. TTS engines are needed for an audio output of machine translation results. Pyttsx is platform independent that is it is compatible with Windows, Linux, and MacOS speech library. This offers a great set of functionality and features. The user can set their voice metadata that is information about the data such as gender male or female, pitch of the voice, age, name and language. It supports large set of voices. So to install it, we can use:

```
>>>shell> pip install pyttsx3
```

```
>>>shell>pip install gTTS
```

Chapter 03

SYSTEM DESIGN

3.1. ARCHITECTURE

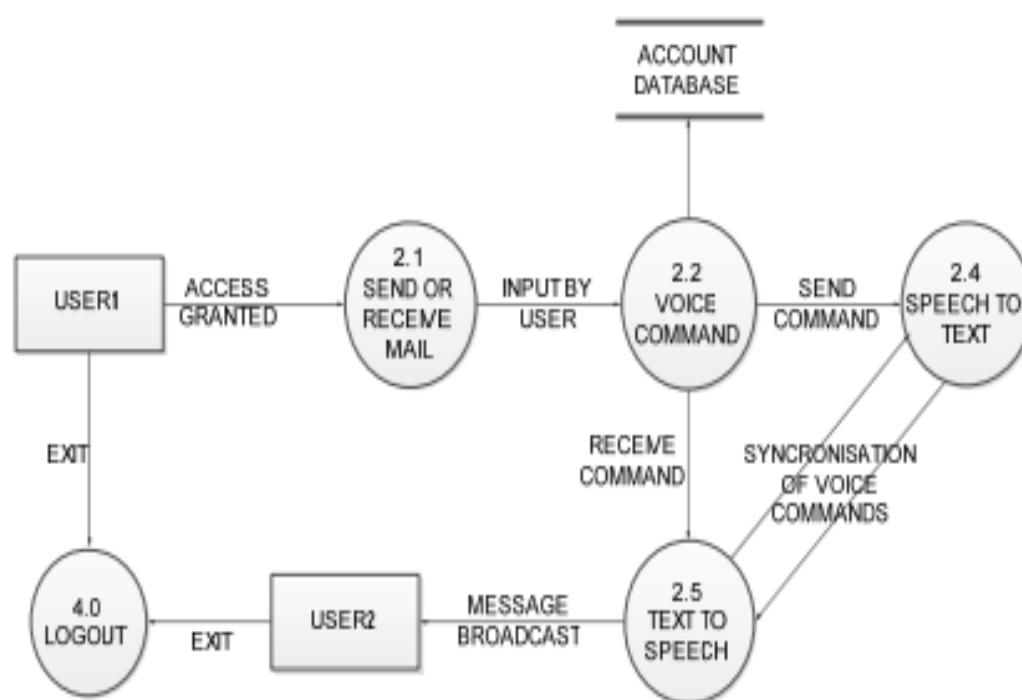


Fig 3.1.1 SYSTEM ARCHITECTURE

The above figure represents the architecture of the above mentioned system. The user1 in the above figure is the visually impaired user who has his/her mail id logged into the system. Therefore the user will be granted access. Later the user will be asked to choose between the options which will be read out to the user using the text-to-speech algorithm. These voice commands are stored into a database. The system now waits for the user's choice and listens to the user through the default microphone or the microphone included in the program. The speech to text and text to speech should be in synchronization with each other to ensure efficient working of the system. Once the choice is taken from the user the required actions are performed. Once the program is done executing the program will stop accessing the Gmail of the user. This way, the

visually challenged can be able to use their accounts and compose mails easily and independently.

3.2. ALGORITHM

The above proposed system will use different technologies such as the text to speech, speech to text, speech recognition and Interactive Voice Response to execute effectively. The algorithm given below gives a clear understanding of the flow of the program

- Start
- A voice message will read out the options to the user using TTS(Text To Speech).
 1. Compose a mail
 2. Check your inbox
- Take the input from the user and convert it into text using STT(Speech To Text)
- If text == '1'
 1. Ask the user for the message to be sent using TTS
 2. Take the input using STT
 3. Send the mail
 4. Print mail sent successfully
- If text == '2'
 - Display and read the number of mails in the inbox.
 - Read from mail-id, subject and body of the most recent mail.
- Logout from the mail
- Exit

3.3. FLOW CHART

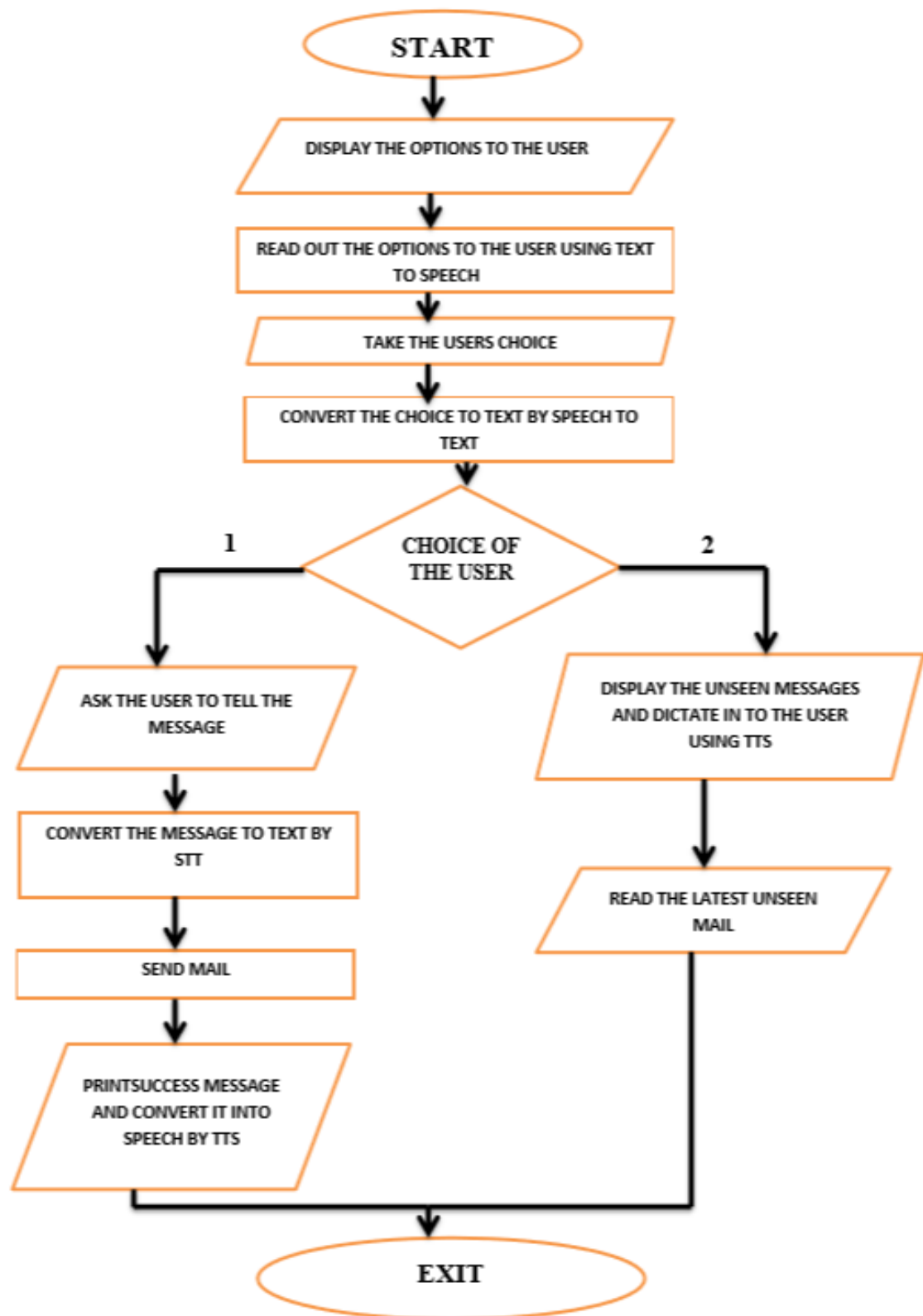


Fig 3.1.2 FLOWCHART OF THE PROGRAM

3.4. CODE

```
import speech_recognition as sr

import smtplib

import pyaudio

import platform

import sys

from bs4 import BeautifulSoup

import email

import imaplib

from gtts import gTTS

import pygame

import os, time

print("-" * 60)

print("    Project: Voice based Email for blind")

print("    <--Created by -->")

print("-" * 60)

# pygame.lib.load_library('avbin')

# pygame.have_avbin=True

# project name

tts = gTTS(text="Project: Voice based Email for blind", lang='en')

ttsname = ("name.mp3")
```

```
tts.save(ttsname)

music = pyglet.media.load(ttsname, streaming=False)

music.play()

time.sleep(music.duration)

os.remove(ttsname)

# login from os

login = os.getlogin

print("You are logging in from : " + login())

# choices

print("1. compose a mail.")

tts = gTTS(text="option 1. compose a mail.", lang='en')

ttsname = ("hello.mp3")

tts.save(ttsname)

music = pyglet.media.load(ttsname, streaming=False)

music.play()

time.sleep(music.duration)

os.remove(ttsname)

print("2. Check your inbox")

tts = gTTS(text="option 2. Check your inbox", lang='en')

ttsname = ("hello.mp3")

tts.save(ttsname)

music = pyglet.media.load(ttsname, streaming=False)
```

```
music.play()

time.sleep(music.duration)

os.remove(ttsname)

# voice recognition part

r = sr.Recognizer()

with sr.Microphone() as source:

    r.adjust_for_ambient_noise(source, duration=5)

    print("Your choice:")

    tts = gTTS(text="Your choice ", lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pygamelet.media.load(ttsname, streaming=False)

    music.play()

    time.sleep(music.duration)

    os.remove(ttsname)

    audio = r.listen(source)

    print("ok done!!")

    tts = gTTS(text="ok done ", lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pygamelet.media.load(ttsname, streaming=False)

    music.play()
```

```
time.sleep(music.duration)

os.remove(ttsname)

try:

    text = r.recognize_google(audio)

    print("You said : " + text)

    tts = gTTS(text="you said "+text, lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pygamelet.media.load(ttsname, streaming=False)

    music.play()

    time.sleep(music.duration)

    os.remove(ttsname)

except sr.UnknownValueError:

    print("Google Speech Recognition could not understand audio.")

    tts = gTTS(text="Google Speech Recognition could not understand audio. ", lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pygamelet.media.load(ttsname, streaming=False)

    music.play()

    time.sleep(music.duration)

    os.remove(ttsname)

except sr.RequestError as e:
```

```
print("Could not request results from Google Speech Recognition service;  
{0}".format(e))
```

```
# choices details
```

```
if text == '1' or text == 'one' or text == 'won' or text == 'on':
```

```
    r = sr.Recognizer() # recognize
```

```
    with sr.Microphone() as source:
```

```
        r.adjust_for_ambient_noise(source, duration=5)
```

```
        print("Your message :")
```

```
        tts = gTTS(text=" your message", lang='en')
```

```
        ttsname = ("hello.mp3")
```

```
        tts.save(ttsname)
```

```
        music = pyglet.media.load(ttsname, streaming=False)
```

```
        music.play()
```

```
        time.sleep(music.duration)
```

```
        os.remove(ttsname)
```

```
        audio = r.listen(source)
```

```
        print("ok done!!")
```

```
        tts = gTTS(text="ok done ", lang='en')
```

```
        ttsname = ("hello.mp3")
```

```
        tts.save(ttsname)
```

```
        music = pyglet.media.load(ttsname, streaming=False)
```

```
music.play()

time.sleep(music.duration)

os.remove(ttsname)

try:

    text1 = r.recognize_google(audio)

    print("You said : " + text1)

    tts = gTTS(text=" You said"+text1, lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pyglet.media.load(ttsname, streaming=False)

    music.play()

    time.sleep(music.duration)

    os.remove(ttsname)

    msg = text1

except sr.UnknownValueError:

    print("Google Speech Recognition could not understand audio.")

    tts = gTTS(text="Google Speech Recognition could not understand audio. ",
lang='en')

    ttsname = ("hello.mp3")

    tts.save(ttsname)

    music = pyglet.media.load(ttsname, streaming=False)

    music.play()
```

```
time.sleep(music.duration)

os.remove(ttsname)


except sr.RequestError as e:

    print("Could not request results from Google Speech Recognition service;
{0}".format(e))

mail = smtplib.SMTP('smtp.gmail.com', 587) # host and port area

mail.ehlo() # Hostname to send for this command defaults to the FQDN of the local
host.

mail.starttls() # security connection

mail.login('!
, '

mail.sendmail('!
) # send
part

print("Congrats! Your mail has been sent. ")

tts = gTTS(text="Congrats! Your mail has been sent. ", lang='en')

ttsname = ("send.mp3")

tts.save(ttsname)

music = pyglet.media.load(ttsname, streaming=False)

music.play()

time.sleep(music.duration)

os.remove(ttsname)

mail.close()
```

```
if text == '2' or text == 'tu' or text == 'two' or text == 'Tu':
```

```
    mail = imaplib.IMAP4_SSL('imap.gmail.com', 993) # this is host and port area.... ssl
    security
```

```
    unm = ('') # username
```

```
    psw = (':') # password
```

```
    mail.login(unm, psw) # login
```

```
    stat, total = mail.select('Inbox') # total number of mails in inbox
```

```
    print("Number of mails in your inbox :" + str(total))
```

```
    tts = gTTS(text="Total mails are :" + str(total), lang='en') # voice out
```

```
    ttsname = ("total.mp3")
```

```
    tts.save(ttsname)
```

```
    music = pygamelet.media.load(ttsname, streaming=False)
```

```
    music.play()
```

```
    time.sleep(music.duration)
```

```
    os.remove(ttsname)
```

```
    # unseen mails
```

```
    unseen = mail.search(None, 'UnSeen') # unseen count
```

```
    print("Number of UnSeen mails :" + str(unseen))
```

```
    #tts = gTTS(text="Your Unseen mail :" + str(unseen), lang='en')
```

```
    #ttsname = ("unseen.mp3")
```

```
    #tts.save(ttsname)
```

```
    #music = pygamelet.media.load(ttsname, streaming=False)
```



```
#music.play()

#time.sleep(music.duration)

#os.remove(ttsname)

# search mails

result, data = mail.uid('search', None, "ALL")

inbox_item_list = data[0].split()

new = inbox_item_list[-1]

old = inbox_item_list[0]

result2, email_data = mail.uid('fetch', new, '(RFC822)') # fetch

raw_email = email_data[0][1].decode("utf-8") # decode

email_message = email.message_from_string(raw_email)

print("From: " + email_message['From'])

print("Subject: " + str(email_message['Subject']))

tts = gTTS(text="From: " + email_message['From'] + " And Your subject: " +
str(email_message['Subject']), lang='en')

ttsname = ("mail.mp3")

tts.save(ttsname)

music = pygame.media.load(ttsname, streaming=False)

music.play()

time.sleep(music.duration)

os.remove(ttsname)

# Body part of mails
```

```
stat, total1 = mail.select('Inbox')

stat, data1 = mail.fetch(total1[0], "(UID BODY[TEXT])")

msg = data1[0][1]

soup = BeautifulSoup(msg, "html.parser")

txt = soup.get_text()

print("Body :" + txt)

tts = gTTS(text="Body: " + txt, lang='en')

ttsname = ("body.mp3")

tts.save(ttsname)

music = pygamelet.media.load(ttsname, streaming=False)

music.play()

time.sleep(music.duration)

os.remove(ttsname)

mail.close()

mail.logout()
```

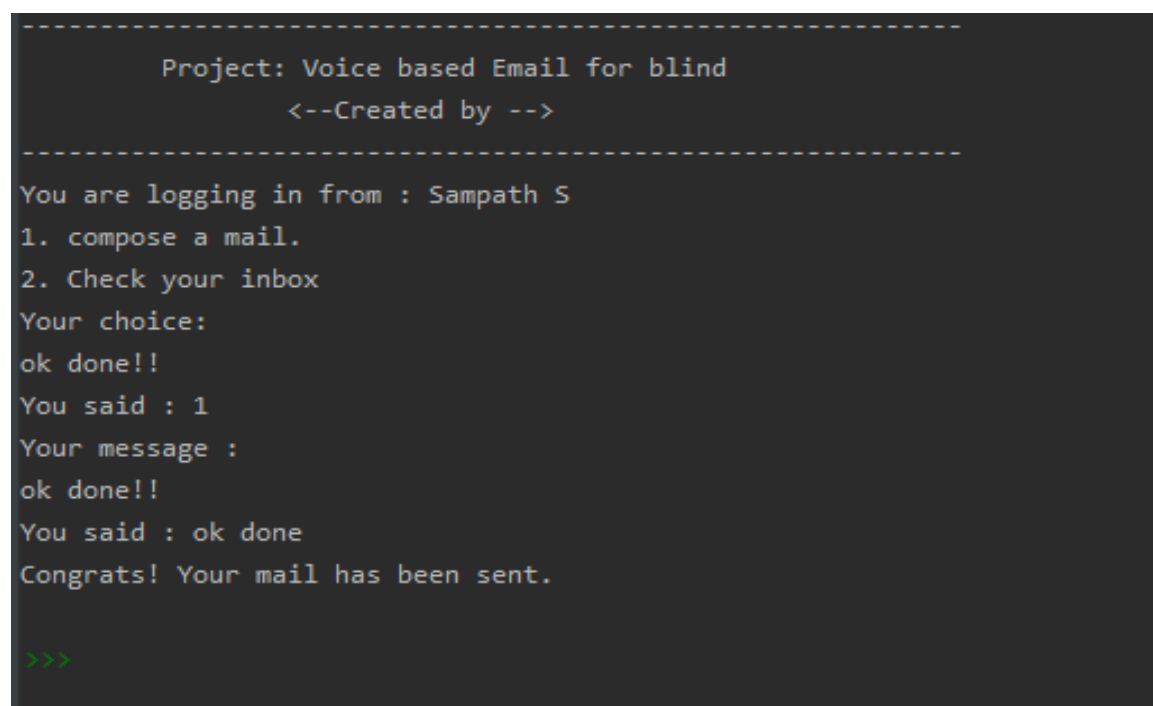
Chapter 04

RESULTS AND DISCUSSIONS

4.1. SUMMARY OF RESULT OBTAINED

When the above proposed system is executed the supplication name will be read out to the user. It can be seen that account that is used to access the operating system is also displayed. The user will be given with the options of composing a mail and checking the inbox which will be dictated to the user. According to the user's choice, either a mail can be composed or the inbox can be checked. Once the user gives his choice the system confirms the choice before proceeding. If the user wants to compose a mail he should speak the message that is to be sent. The message will be read out to the user once before sending the mail. If the user opts for checking the inbox then the number of mails in the inbox, the unseen mails will be dictated and the latest unseen message will be read along with the from email-id, the subject of the mail and its body.

4.2. OUTPUT



```
-----  
Project: Voice based Email for blind  
  <--Created by -->  
-----  
You are logging in from : Sampath S  
1. compose a mail.  
2. Check your inbox  
Your choice:  
ok done!!  
You said : 1  
Your message :  
ok done!!  
You said : ok done  
Congrats! Your mail has been sent.  
  
>>>
```

Fig 4.2.1 OUTPUT WHEN THE USER OPTS OPTION 1

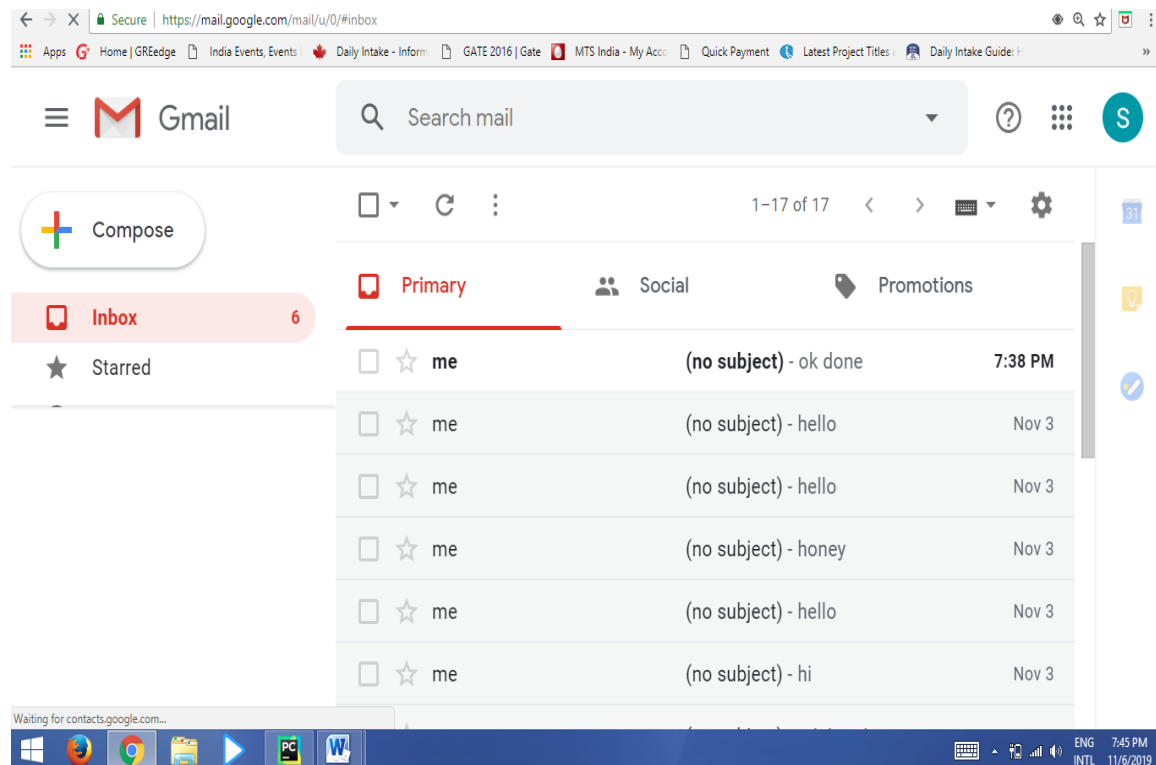


Fig 4.2.2 THE MAIL HAS BEEN SENT SUCCESSFULLY

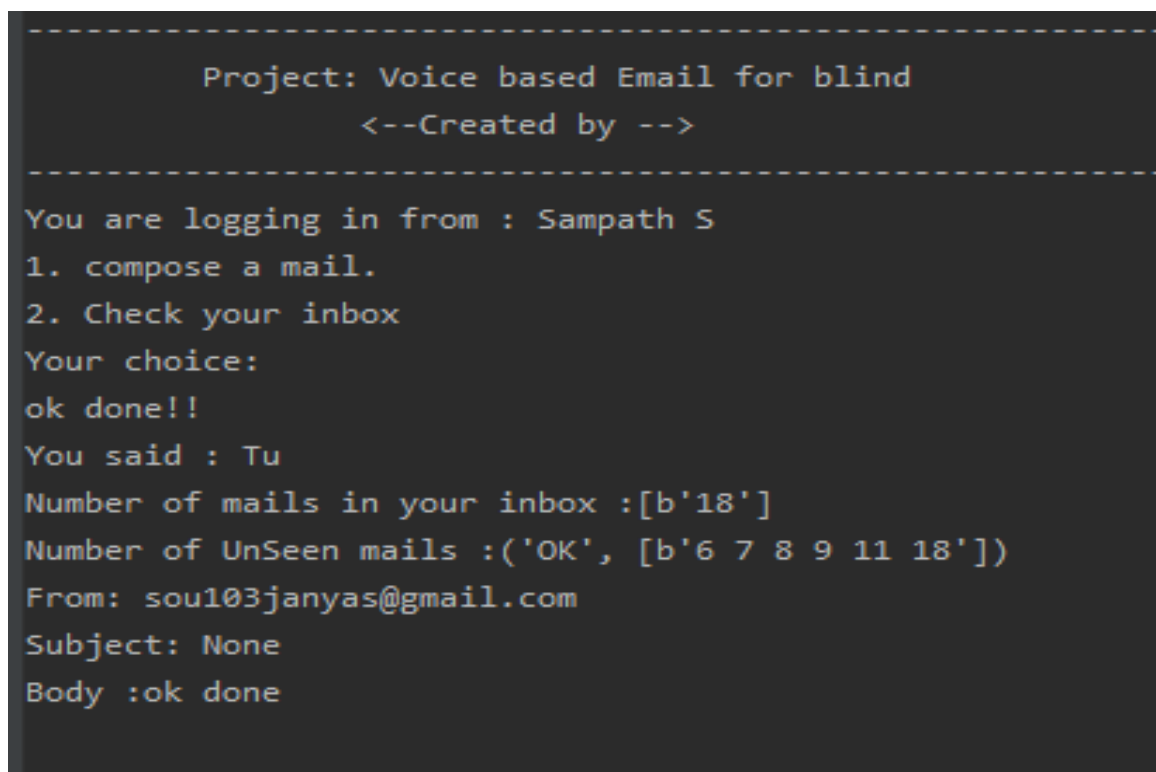


Fig 4.2.3 OUTPUT WHEN THE USER OPTS FOR OPTION 2

Chapter 05

CONCLUSION

The voice based email for the blind proves to be an application that can be used by the visually challenged as well as the illiterate as this system not only eliminates the need for reading the contents on the screen and also the need of mouse or a keyboard. The user can now send emails without depending on anyone through voice commands. This e-mail system can be used by any user of any age group with ease of access. It has feature of speech to text as well as text to speech with speech reader which makes designed system to be handled by visually impaired person as well as blind person. Voice could be extended to image attachments and other options such as indentation, fonts etc., that are available with normal E-Mail.

REFERENCES

- <https://www.seminaronly.com/Engineering-Projects/Computer/voice-based-email-system.php>
- [http://122.252.232.85:8080/jspui/bitstream/123456789/16961/1/SP13277 Saurav%20Mishra 141318 Ashwani%20Panwar 141294 CSE 2018.pdf](http://122.252.232.85:8080/jspui/bitstream/123456789/16961/1/SP13277_Saurav%20Mishra_141318_Ashwani%20Panwar_141294_CSE_2018.pdf)
- <https://www.arcjournals.org/pdfs/ijrscse/v3-i1/5.pdf>
- <https://www.ijariit.com/manuscripts/v3i3/V3I3-1462.pdf>
- <https://www.geeksforgeeks.org/project-idea-voice-based-email-visually-challenged/>
- <https://github.com/hacky1997/voice-based-email-for-blind/blob/master/requirements.txt>
- <https://www.geeksforgeeks.org/speech-recognition-in-python-using-google-speech-api/>
- <https://mail.google.com/mail/u/0/#inbox>