

VOICE-BASED EMAIL SYSTEM FOR VISUALLY IMPAIRED

A Project Report

Submitted to the Faculty of Engineering of
**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA,
KAKINADA**

In partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE AND ENGINEERING

By

SIRAZUNNISA
(18481A05K2)

V. SAI CHAND
(18481A05M9)

Y. HIMAJA
(18481A05N9)

Under the guidance of
Dr. Y. Adilakshmi, MTech, Ph.D
Associate Professor, Department of CSE



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SESHADRIRAO GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
SESHADRIRAO KNOWLEDGE VILLAGE
GUDLAVALLERU – 521356
ANDHRA PRADESH
2021-2022

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE, GUDLAVALLERU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project report entitled “**VOICE-BASED EMAIL SYSTEM FOR VISUALLY IMPAIRED**” is a bonafide record of work carried out by **SIRAZUNNISA (18481A05K2), V.SAI CHAND (18481A05M9), Y. HIMAJA (18481A05N9)** under the guidance and supervision in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Jawaharlal Nehru Technological University Kakinada, Kakinada during the academic year 2021-22.

Project Guide
(Dr. Y. Adilakshmi)

Head of the Department
(Dr. M. BABU RAO)

External Examiner

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragements crown all the efforts with success.

We would like to express our deep sense of gratitude and sincere thanks to **Dr. Y. Adilakshmi, Associate Professor**, Department of Computer Science and Engineering for her constant guidance, supervision, and motivation in completing the project work.

1

We feel elated to express our floral gratitude and sincere thanks to **Dr. M. Babu Rao**, Head of the Department, Computer Science and Engineering for his encouragement all the way during the analysis of the project. His annotations, insinuations, and criticisms are the key to the successful completion of the project work.

We would like to take this opportunity to thank our beloved principal **Dr. G.V.S.N.R.V Prasad** for providing great support for us in completing our project and giving us the opportunity for doing a project.

Our Special thanks to the faculty of our department and the programmers of our computer lab. Finally, we thank our family members, non-teaching staff, and our friends, who had directly or indirectly helped and supported us in completing our project in time.

Team members

SIRAZUNNISA (18481A05K2)
V.SAI CHAND (18481A05M9)
Y. HIMAJA (18481A05N9)

INDEX

TITLE	PAGENO
LIST OF FIGURES	I
LIST OF ABBREVIATIONS	II
ABSTRACT	III
CHAPTER 1: INTRODUCTION	1
1.1 INTRODUCTION	1
1.2 OBJECTIVES OF THE PROJECT	2
1.3 PROBLEM STATEMENT	2
CHAPTER 2: LITERATURE REVIEW	4
2.1 LITERATURE REVIEW	4
2.2 EXISTING SYSTEMS	6
CHAPTER 3: PROPOSED SYSTEM	7
3.1 METHODOLOGY	7
3.1.1 FLOW CHART	7
3.1.2 SPEECH RECOGNITION	8
3.1.3 TEXT TO SPEECH	16
3.1.4 SMTPLIB	17
3.1.5 EASYIMAP	19
3.1.6 PSEUDO CODE	20
3.2 IMPLEMENTATION	21
3.2.1 EXECUTION STEPS	21
3.3 DATA PREPARATION	26
CHAPTER 4: RESULTS AND DISCUSSION	27
CHAPTER 5: CONCLUSION AND FUTURE SCOPE	31
BIBILOGRAPHY	32

LIST OF FIGURES

FIG NO	FIGURE	PAGE NO
3.1	FLOW CHART	7
3.2	Speech to Text Conversion	10
3.3	Error message for unknown input	14
3.4	Text to Speech	16
3.5	Listen Function	22
3.6	Speak Function	23
3.6	Readmail Function	24
3.8	Sendmail Function	25
3.9	Voice Instructions were given to the user	26
4.1	Initial Voice Instructions	27
4.2	When the user select send option	28
4.3	Mail is sent to a receiver	28
4.4	When the user selects read-option	29
4.5	Read Mail image in Gmail	29
4.6	When the user selects exit-option	30

LIST OF ABBREVIATIONS

ABBREVIATION	EXPLANATION
NLP	Natural Language Processing
TTS	Text-To-Speech
STT	Speech-To-Text
IVR	Interactive Voice Response
HMM	Hidden Markov Model
VAD	Voice Activity Detectors
SMTP	Simple Mail Transfer Protocol

ABSTRACT

The Internet is the most essential part of today's world of communication. E-mails are a further important way of communication that is widely used in the business world. This technology has been useless for incapacitated and blind people. There are around 260 million visually challenged people around the globe and to make this E-mail system closer to visually challenged people, A Voice-Based E-mail System has been proposed. This system provides them the facility of communication by using natural language processing (NLP) packages to convert text-to-speech (TTS) and speech-to-text (STT) so that visually impaired people can operate the system easily. This system reduces the complexity to memorize the characters or information regarding keyboard shortcuts. Every function will be based on simple voice commands so that those people can easily make use of the technologies.

Keywords: E-mails, Natural Language Processing, Speech-to-text, Text-to-Speech, Blind People

CHAPTER – 1

INTRODUCTION

1.1 INTRODUCTION:

The internet has become one of the most important tools for everyday life. With the evolution of numerous technologies utilizing the internet, communication has become very simple, and all works can be completed in a shorter time period with more precision and efficiency. Communication is one of the disciplines that has advanced to another level as a consequence of technical breakthroughs and the internet's accessibility. Because of technology improvements, communication has become so simple that distance is no longer an issue.

The first thing that comes to mind when we think of online communication, especially for commercial purposes, is email communication. Electronic Mail or simply Email is defined as a way of exchanging information among people using electronic devices such as computers, mobile phones, tablets, etc. Email is one of the most reliable and extensively used techniques for sending important information; but, in order to use the internet, one should not be blind. There are some visually challenged people among us, who are not able to see and thus cannot have access to the computer screen or keyboard.

Millions of people in this world who are blind or visually impaired are unable to see the screen or use the keyboards. There are more than 250 million visually challenged people around the globe. That is, around 250 million people are unaware of how to use the Internet or E-mail. The only way by which a visually impaired person can send an E-mail is, they have to dictate the entire content of the mail to a third person (not visually challenged) and then the third person will compose the mail and send it on their behalf of the visually impaired person.

But this is not the correct way to deal with this problem. It is very less likely that every time a visually challenged person can find someone for help. Although for these reasons the specially-abled people are criticized by our society. Hence the existing email system is worthless for persons who are visually challenged.

Visually impaired and blind persons find keyboard typing challenging because they must rely on others to do so. Because a person must be able to see and read what is printed on the screen in order to access the internet, it is a useless technology for visually

impaired persons. There is only one way for a visually impaired person to send an email: They must provide the entire email content to a third party so that the third party can compose and send it on the visually impaired person's behalf. However, this method does not lead to a solution to the problem. For maintaining the Integrity of the Specifications, a visually impaired person cannot always find a third person, and the content can sometimes be personal. The optimum solution to this issue is speech-to-text conversion. Speech recognition can be beneficial to a wide range of people.[4]

As a result, we designed a voice-based email system for the visually impaired, which will assist blind people in sending and reading emails. Users of this system do not need to memorize any basic information about keyboard shortcuts or key locations because our system relies solely on the user's voice commands and does not use keyboards. Using text-to-speech and speech-to-text conversions, our project aims to make the email system familiar to visually impaired persons. Text-to-speech technology turns text into human-sounding speech and allows you to build your distinctive voice for usage in your apps. Speech-to-text turns the user's voice into text.

1.2 OBJECTIVES OF THE PROJECT:

- Aims to develop an email system that will help even a naïve, visually impaired person to use the services for communication without previous training
- Doesn't require the use of a keyboard.
- User-Friendly for any kind of person
- Can operate only by using voice commands

1.3 PROBLEM STATEMENT:

Emails are an important way of communication, especially in business communications but this system is useless for blind people as the existing system doesn't provide any voice assistance to them. due to this, blind or visually impaired people are unable to use this email system.

And in most cases, people suggest using a braille keyboard for blind people, those braille keyboards are useful but costly and difficult to understand. And another one is

screen readers which are used to read the contents on the screen but they are not useful to send and receive e-mails they just convert text to speech but screen readers do not convert speech to text.

Our Project provides a voice-based email system for visually impaired people so that they can send and read emails using voice commands and can operate without the use of keyboards. In the proposed project, the voice prompt guides the user and asks if the user wants to send or read the mail, and this voice guides until the user asks to exit.

CHAPTER – 2

LITERATURE REVIEW

2.1 LITERATURE REVIEW

[1] Mamatha, A., Jade, V., Saravana, J., Purshotham, A., & Suhas, A. V in “Voice based email system for blind” mentioned that to overcome the disadvantages of classical ASR and screen reading systems, a voice-based email system was proposed recently in 2020. The system has advanced capabilities that allow blind people to operate it with convenience. The first module is the Login module, which examines the login credentials. After registering in, the client is taken to the home module, where the following options are available: Inbox, Create, Sent mail, and Junk. In PC program design, IVR technology is used, along with STT (Speech-to-message) and TTS (Text-to-discourse). Mouse click events are also used in the proposed system.

[2] The authors Belekar, A., Sunka, S., Bhawar, N., & Bagade proposed a voice-based email system in the paper, whereby they integrated Google's Gmail. Traditional systems had their email services that have been constructed by the users. The module includes (a) a Speech-to-Text Converter and (b) a Speech-to-Text Converter. A text-to-speech system is a device that adherent's text into speech. For sending emails, the application uses the SMTP protocol, and for receiving emails, it uses the POP3 protocol. Speech-to-text accuracy is low since it needs to be educated. It is a desktop application that may be used by people who are illiterate or disabled. The suggested system not only protects the security of the user's data but also gives a feeling of secure mailing to the user.

[3] The researchers of the paper Khan, R., Sharma, P. K., Raj, S., Verma, S. K., & Katiyar, S proposed an email system that is simple to use for visually impaired people. TTS (Text-to-speech), STT (Speech-to-text), and Mail Software Program (Collaborating, Mailbox, and Then sent Mail) are the three modules that constitute the system design. Speech-to-text is conducted in this system using Artificial Intelligence (AI) via an API involving neural network models given by Google Cloud Speech-to-text to developers. Furthermore, it generates credentials or other credentials into cryptographic algorithms using various Hashing Algorithms (MD5, SHA), resulting in stronger security than older iterations.

[5] The paper by Payal Dudhbale J. S.Wankhade, P. S. Narawade proposes a system that relies on a system with a voice command, based on that, in contrast to the pre-existing email system. The whole system is, in essence, is based on converting the number to text. Once made use, the implementation of the system will prompt the user to speak commands to make use of the relevant services. If the user wants to access the relevant services, it is necessary to state that this command will work. This program uses the IMAP (Internet Message Access Protocol). This is a standard Internet protocol used by an email to send an email from a mail server over TCP / IP. The main type of activity, the screen, will be the first screen displayed from the beginning of the year. On this screen, waiting for the user to press a single button, the system will start to receive your voice commands. It's a full-size single button to tap anywhere on the screen. Then, with the help of voice commands, the user can send an email to read it.

[6] The system uses three main technologies:

- To convert the number to text
- Text-to-speech.
- Interactive Voice Response.

When you are visiting the site for the first time, it needs to register with the help of voice commands. Also, once you register, the user's audio data and a note will be saved in the database. And the user will receive a user id and password after the user logs in to receive an email in such a system. The user interface has been developed with the help of Adobe Dreamweaver CS3. The site is primarily centered on the concept of efficiency and effectiveness. In addition, there is a "Contacts" page, where the user can offer any suggestions or any help if they need it.

[7] At the time, one of the e-system is proposed by G. Shoba, G. Anusha, V.Jeevitha, R.Shanmathi, which is easily accessible for the blind. You can use the voice-to-text converter-text-to-speech converter and the Viterbi algorithm. The algorithmic rule, which is working with the technology, does not find it to be the most appropriate word; as soon as the user says so, so it is, as your guessed word, for a given the word pronounced. The user registers at the site where they are for the first time, the visit to the site. This system will reduce some of the disadvantages of the current system. Sorry, this scheme is the efficiency of the Viterbi algorithm to reduce the number of errors will increase and require more space.

[4] In, a system proposed by Rijwan Khan, Pawan Kumar sharma, Sumit Raj, Sushil Kr. Verma, Sparsh Katiyar for the blind and the illiterate is proposed to improve their interaction with the email system. This system eliminates the use of IVR technology that are using screen readers, Braille keyboards. There, we used the speech-to-text and text-to-speech conversion. Voice commands are also used for other activities as well. For registration, you may use your identity, your email address, and your password. This is functionality to use the function that tells PHP to email. This is the library, which you can use to send an email. To obtain the user's email from the IMAP server. This Lash-Morris-Pratt algorithm is used to search for email collection boxes. Thus, the system's environment is clean, the voice is controlled by a feedback system in each step. Sorry, this scheme is that it uses Gmail as a host server so that we can make use of other email services like Yahoo, Google, etc.

2.2 EXISTING SYSTEM

On analyzing similar proposals and research papers regarding voice-based email, the technologies commonly used in those systems are

[8] Screen Readers: Screen readers are software programs that allow the users to read the contents on the screen using a speech synthesizer. It is an interface between a computer's operating system and its application and the user. It makes the user enter the commands that should be said by the speech synthesizer. The user can locate the cursor, focus on the text, read a location on the map and many more activities can be performed.

[9] Braille Keyboards: Braille computer keyboards are very rare. They are connected to a computer similar to a standard computer keyboard. It accepts the input from it. They are also used in Braille Typewriters.

[10] IVR: It is an Interactive Voice Response system. It allows the user to interact with the system through voice. It analyses and synthesizes the user's voice and replies back to them in the form of voice, text, email, etc. Speech to text conversion: The given voice input is taken to the server where it gets converted as text output. Text to speech conversion: The work is similar to the previous one but the output response will be in the form of voice. The disadvantages of the existing system are as follows:

1. Screen readers cannot spell technical and biological terms.
2. Braille language must be known and when using keyboards shortcuts needs to be notified.
3. Fingerprint authentication can be easily acquired through any malpractices.

CHAPTER – 3

PROPOSED METHOD

3.1 METHODOLOGY

Emails are a significant means of communication, mainly in business, but they are worthless for blind people since the current system does not give any voice assistance. As a result, those who are blind or visually challenged are unable to access this email system.

This Voice based email system solves this problem by facilitating blind people to send and read emails only using voice commands and with no use of keyboards. This Voice based email system completely uses the NLP subfield packages and here we are using mainly four modules. These modules help to recognize the user's voice and convert the voice to speech and also convert speech to text and make the user send and read emails.

The main modules used in the proposed system are:

- Speech Recognition
- Pyttsx3
- Smtplib
- Easyimap

3.1.1 FLOW CHART

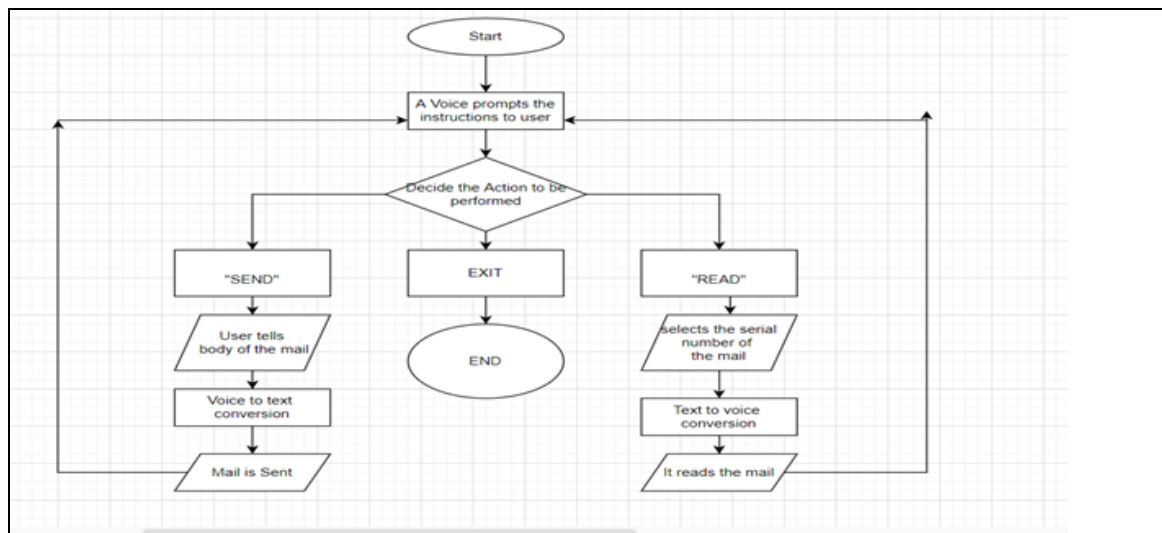


FIGURE 3.1 FLOW CHART

The above flowchart shows the execution of this proposed system. when we run this application, a voice prompts the instructions to the user. It prompts them to choose the action which they want to whether it may be read, send, or exited. When the user

chooses to send the mail, the voice asks the user to give the details of the mail like subject and body of the mail and audio given by the user is converted into a textual format using speech recognition module and then the mail will be sent and when the user chooses to read the mail, the voice asks the serial number of the mail which they want to read and then the pyttsx3 module converts the text in the mail to the voice format so that the blind person can easily acknowledge the mail. The voice instructions are continuous in a loop until the user asks to exit.

3.1.2 Speech Recognition

There are many packages available for speech recognition exist on PyPI. A few of them include:

- SpeechRecognition
- watson-developer-cloud
- google-cloud-speech
- apiai
- assemblyai
- pocketsphinx
- wit
- CMU Sphinx

The most common and best package which helps in the speech recognition process is SpeechRecognition.

Speech Recognition uses a combination of linguistics and computer science to recognize spoken words and convert them to textual format. The Speech Recognition module enables computers to comprehend human speech. The ability of a machine to listen to spoken words or audio and recognize them is known as speech recognition. The uttered words can then be converted to text, a query can be made, and a response can be given using Python's speech recognition. Some devices can even be programmed to respond to spoken speech. Using computer programs information is taken from the microphone, process it, and convert it into a proper form, you may accomplish voice recognition in Python.

Speech recognition has its roots in research done at Bell Labs in the early 1950s. Early systems were limited to a single speaker and had limited vocabularies of about a dozen words. Modern speech recognition systems have come a long way since their ancient counterparts. They can recognize speech from multiple speakers and have enormous vocabularies in numerous languages.

The first component of speech recognition is, of course, speech. Speech must be converted from physical sound to an electrical signal with a microphone, and then to digital data with an analog-to-digital converter. Once digitized, several models can be used to transcribe the audio to text.

Most modern speech recognition systems rely on what is known as a Hidden Markov Model (HMM). This approach works on the assumption that a speech signal, when viewed on a short enough timescale (say, ten milliseconds), can be reasonably approximated as a stationary process—that is, a process in which statistical properties do not change over time.

In a typical HMM, the speech signal is divided into 10-millisecond fragments. The power spectrum of each fragment, which is essentially a plot of the signal's power as a function of frequency, is mapped to a vector of real numbers known as cepstral coefficients. The dimension of this vector is usually small—sometimes as low as 10, although more accurate systems may have dimension 32 or more. The final output of the HMM is a sequence of these vectors.

To decode the speech into text, groups of vectors are matched to one or more phonemes—a fundamental unit of speech. This calculation requires training, since the sound of a phoneme varies from speaker to speaker, and even varies from one utterance to another by the same speaker. A special algorithm is then applied to determine the most likely word (or words) that produce the given sequence of phonemes.

One can imagine that this whole process may be computationally expensive. In many modern speech recognition systems, neural networks are used to simplify the speech signal using techniques for feature transformation and dimensionality reduction

before HMM recognition. Voice activity detectors (VADs) are also used to reduce an audio signal to only the portions that are likely to contain speech. This prevents the recognizer from wasting time analyzing unnecessary parts of the signal.

Fortunately, A number of speech recognition services are available for use online through an API, and many of these services offer Python SDKs. Some of these packages—such as wit and apiai—offer built-in features, like natural language processing for identifying a speaker’s intent, which go beyond basic speech recognition. Others, like google-cloud-speech, focus solely on speech-to-text conversion.

Speech recognition may appear futuristic, but it is already in use. Automated phone calls allow you to shout out your question or the question for which you would like assistance; virtual assistants such as Siri or Alexa also use speech recognition to converse with you.

Python uses linguistic and acoustic modeling methods to accomplish speech recognition. Acoustic modeling is a technique for identifying phonemes/phonetics in a speech to extract the most important parts of speech, such as words and sentences.

With the use of a microphone, speech recognition begins by transforming the sound energy produced by the person speaking into electrical energy. The electrical energy is then converted from analog to digital, and finally to text.

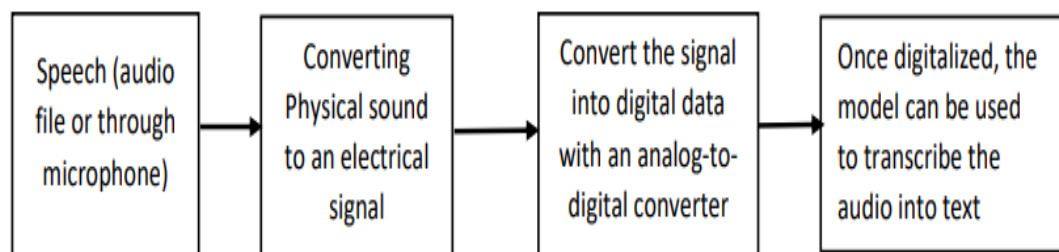


FIGURE 3.2 : SPEECH TO TEXT CONVERSION

It takes the audio data and breaks it down into sounds, then uses algorithms to analyze the sounds to identify the most likely word that fits the audio. Natural Language Processing and Neural Networks are used to do all of this. Hidden Markov models can be used to enhance accuracy by detecting temporal patterns in speech.

The SpeechRecognition library acts as a wrapper for several popular speech APIs and is thus extremely flexible. One of these—the Google Web Speech API—supports a default API key that is hard-coded into the SpeechRecognition library. That means you can get off your feet without having to sign up for a service.

The flexibility and ease of use of the SpeechRecognition package make it an excellent choice for any Python project. SpeechRecognition will work out of the box if all you need to do is work with existing audio files. Specific use cases, however, require a few dependencies. Notably, the PyAudio package is needed for capturing microphone input.

Installation of Speech Recognition Module:

Installation of the SpeechRecognition library with pip, run the following command

➤ `pip install SpeechRecognition`

Installing PyAudio:

PyAudio is a free, open-source library to play and record audio files on various platforms. This library works as an extension of PortAudio. PyAudio will allow microphone access from the laptop or computer or any device.

Installation of the PyAudio library with pip, run the following command

➤ `pipwin install pyaudio`

Recognizer Class

For APIs, the SpeechRecognition library acts as a wrapper and it is extremely flexible and compatible. The SpeechRecognition library has several libraries but we will only be focusing on the Recognizer class. The Recognizer class will help us to convert the audio data into text files. So, now let's try out the SpeechRecognition library, and the next step is simple we just need to install it in your environment.

let the magic start with the Recognizer class in the SpeechRecognition library. The main purpose of a Recognizer class is of course to recognize speech. Creating a Recognizer instance is easy we just need to type:

➤ `recognizer = sr.Recognizer()`

Using listen() to Capture Microphone Input

Just like the AudioFile class, Microphone is a context manager. You can capture input from the microphone using the listen() method of the Recognizer class inside of the with block. This method takes an audio source as its first argument and records input from the source until silence is detected.

➤ `with Microphone as source:`

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

Once you execute the with block, try speaking something into your microphone. Wait a moment for the interpreter prompt to display again. Once the prompt returns, you're ready to recognize the speech.

If the prompt never returns, your microphone is most likely picking up too much ambient noise. You can interrupt the process with Ctrl+C to get your prompt back.

After completing the installation process let's set the energy threshold value. You can view the energy threshold value as the loudness of the audio files the ideal energy threshold value is 300. The documentation of SpeechRecognition recommended 300 values as a threshold and it works best with various audio files.

Using the energy threshold will improve the recognition of speech while working with audio data. If the values are higher than the energy threshold = 300 then are considered speech but if the values are lower than they are considered silent.

➤ `recognizer.energy_threshold = 300`

Speech Recognition Functions

For recognizing speech from audio data using different APIs there is a recognizer class that does all the work.

- `recognize_houndify()`: Houndify by SoundHound
- `recognize_ibm()`: IBM Speech to Text
- `recognize_sphinx()`: CMU_Sphinx – requires installing PocketSphinx
- `recognize_google()`: Google_Web_Speech_API
- `recognize_google_cloud()`: Google Cloud Speech – requires installation of the google-cloud-speech package

The `recognize_sphinx()` has benefits as it can work offline with the CMU Sphinx engine. The other requires a stable internet connection.

Google offers its own API `recognize_google()` which is free and it also does not require any API key for use. Well, there is one drawback about Google speech recognition that is limiting you when you try to process the audio data which have a longer time period.

The Effect of Noise on Speech Recognition

We won't get noise-free data every time. All audio files have some degree of noise in them from the start and this un-handled noise will affect the accuracy of the Speech Recognition system.

To resolve this issue, we can use the `adjust_for_ambient_noise` method of the Recognizer class. The `adjust_for_ambient_noise()` method reads the first second of the file stream and calibrates the recognizer to the noise level of the audio. Hence, that portion of the stream is consumed before you call `record()` to capture the data. `adjust_for_ambient_noise()` analyzes the audio source for one second. If this seems too long to you, feel free to adjust this with the duration keyword argument.

You can adjust the time frame that `adjust_for_ambient_noise()` uses for analysis with the duration keyword argument. This argument takes a numerical value in seconds and is set to 1 by default. Try lowering this value to 0.5. The SpeechRecognition documentation recommends using a duration no less than 0.5 seconds. In some cases, you may find that

durations longer than the default of one second generate better results. The minimum value you need depends on the microphone's ambient environment

Handling Unrecognizable Speech

When you run a code and make some unintelligible noises into the microphone. You should get something like this in response:

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/david/real_python/speech_recognition_primer/venv/lib/python3.5/site-pack
    if not isinstance(actual_result, dict) or len(actual_result.get("alternative", []
speech_recognition.UnknownValueError
```

FIGURE:3.3 Error message when input is unknown

Audio that cannot be matched to text by the API raises an `UnknownValueError` exception. You should always wrap calls to the API with try and except blocks to handle this exception.

Difficulties in developing a speech recognition system

Developing a high-quality speech recognition system is really a difficult problem. The difficulty of speech recognition technology can be broadly characterized along with a number of dimensions as discussed below –

Size of the vocabulary – the size of the vocabulary impacts the ease of developing an ASR. Consider the following sizes of vocabulary for a better understanding.

A small size vocabulary consists of 2-100 words, for example, as in a voice-menu system

A medium size vocabulary consists of several 100s to 1,000s of words, for example, as in a database-retrieval task

A large size vocabulary consists of several 10,000s of words, as in a general dictation task.

Channel characteristics – Channel quality is also an important dimension. For example, human speech contains high bandwidth with full frequency range, while a telephone speech consists of low bandwidth with limited frequency range. Note that it is harder in the latter.

Speaking mode – The ease of developing an ASR also depends on the speaking mode, that is whether the speech is in isolated word mode, connected word mode, or a continuous speech mode. Note that a continuous speech is harder to recognize.

Speaking style – A read speech may be in a formal style, or spontaneous and conversational with a casual style. The latter is harder to recognize.

Speaker dependency – Speech can be speaker-dependent, speaker adaptive, or speaker-independent. A speaker's independence is the hardest to build.

Type of noise – Noise is another factor to consider while developing an ASR. Signal to noise ratio may be in various ranges, depending on the acoustic environment that observes less versus more background noise –

If the signal to noise ratio is greater than 30dB, it is considered a high range

If the signal to noise ratio lies between 30dB to 10db, it is considered a medium SNR

If the signal to noise ratio is lesser than 10dB, it is considered a low range

Microphone characteristics – The quality of the microphone may be good, average, or below average. Also, the distance between the mouth and the micro-phone can vary. These factors also should be considered for recognition systems.

Size of the vocabulary – Size of the vocabulary impacts the ease of developing an ASR. Consider the following sizes of vocabulary for a better understanding.

Supported File Types

Currently, SpeechRecognition supports the following file formats:

- WAV: must be in PCM/LPCM format
- AIFF
- AIFF-C
- FLAC: must be native FLAC format; OGG-FLAC is not supported.

3.1.3 TEXT TO SPEECH

pyttsx3 is a text-to-speech conversion library written in Python. It works offline, unlike other libraries, and in Python 2 and 3 compatibles. An application uses the `pyttsx3.init()` factory method to get a reference to a `pyttsx3`. Engine instance. It's a basic program that converts text into speech.

The `pyttsx3` module has two voices, one female and the other male, both of which are provided by "sapi5" for Windows.

To get started with installing the `pyttsx3` module, open a terminal and type

➤ `pip install pyttsx3`

You'll need to install `pywin32` if you get problems like No module named `in32com.client`, no module named `win32`, or No module named `win32api`.

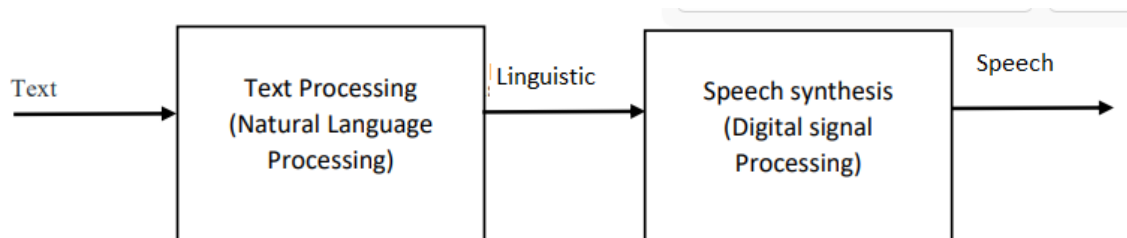


Figure:3.4 Text to Speech conversion using pyttsx3 module

Pyttsx3 Engine Initialization

➤ `engine = pyttsx3.init()`

The above code initializes the `pyttsx3` package. The Instance of the initialized `pyttsx3` package is stored in the `engine` variable. We are calling the variable `engine` as it works as the engine and converts Text-To-Speech whenever execute the functions from the package.

Say Function in pyttsx3

➤ `engine.say("This is Text-To-Speech Engine Pyttsx3")`

There is a built-in `say()` function in the `pyttsx3` package that takes a string value and speaks it out.

runAndWait Function

➤ `engine.runAndWait()`

This function keeps track of when the engine starts converting text to speech and waits for that much time, and does not allow the engine to close. If we don't write this code, it may happen that the engine might not work properly as the processes will not be synchronized. After all the processes are over, we shut down the engine by calling the `stop()` function.

Change Text-to-Speech Voice in Pyttsx3 Python

We can also change the voice of the engine; the default is the voice of a male named David. To change the voice of the `pyttsx3` engine, first, we will have to get the list of objects of voices.

➤ `voices = engine.getProperty('voices')`

The `getProperty()` function of the `pyttsx3` package takes a string as a parameter and returns an object matching the string.

3.1.4 SMTPLIB

SMTP stands for Simple Mail Transfer Protocol, and it is a protocol for sending and routing emails between mail servers. The `smtplib` module in Python creates an SMTP client session object that may be used to deliver mail to any system on the Internet that has an SMTP or ESMTP listener daemon.

Here is the detail of the parameters –

- **host** – This is the host running your SMTP server. You can specify the IP address of the host or a domain name like `tutorialspoint.com`. This is an optional argument.
- **port** – If you are providing a *host* argument, then you need to specify a port, where the SMTP server is listening. Usually, this port would be 25.
- **local_hostname** – If your SMTP server is running on your local machine, then you can specify just *localhost* as this option.

An SMTP object has an instance method called **Sendmail**, which is typically used to do the work of mailing a message. It takes three parameters –

- The *sender* – A string with the address of the sender.
- The *receivers* – A list of strings, one for each recipient.
- The *message* – A message as a string formatted as specified in the various RFCs.

To install smtplib the command is:

➤ `pip install secure-smtplib`

To send the email, connect to the local SMTP server with the `smtpObj` object, and then use the `SendMail` method with the text, from address, and destination address as options (despite the fact that the from and to address are within the e-mail, they are not used to route messages).

If you don't have an SMTP server installed on your local workstation, you can use the `smtplib` client to communicate with a remote SMTP server. Unless you use a webmail service (such as Hotmail or Yahoo! Mail), your e-mail provider should have provided you with the following outgoing mail server information:

➤ `smtplib.SMTP('mail-domain.com', 25)`

A secure connection with Gmail's SMTP server, using the `SMTP_SSL()` of `smtplib` to initiate a TLS-encrypted connection. The default context of SSL validates the hostname and its certificates and optimizes the security of the connection.

Using `smtplib.SMTP_SSL()` as server: makes sure that the connection is automatically closed at the end of the indented code block. If the port is zero, or not specified, `SMTP_SSL()` will use the standard port for SMTP over SSL (port 465).

Sending Your Plain-text Email

After you initiated a secure SMTP connection using either of the above methods, you can send your email using `sendmail()`, which pretty much does what it says on the tin:

➤ `server.sendmail(sender_email, receiver_email, message)`

3.1.5 EASYIMAP

Easyimap is a simple IMAP wrapper. It is used to read and send emails using python. We can also use the IMAP library, but it's a long and difficult process, so we'll use easyimap instead, which is simple and quick to install. To install easyimap, enter the command below in your terminal.

➤ `Pip install easyimap`

To connect to the IMAP server, first, build an object in which we must specify which server we want to connect to; in this case, we are connecting to the Gmail server.

➤ `Server_obj = e.connect("imap.gmail.com",username, password)`

To list the ID of the email in our unread inbox, we use **`server.listids`** and we can create a variable to acquire the email's context, and we can also specify the exact number of the email.

Some of the Imapper methods are:

- `listids(limit=10)`→ Returns list of available email ids.
- `listup(limit=10)`→Returns list of tuples(email_id, mail_object).
- `unseen(limit=10)`→Returns list of types(email_id, mail_object).

- mail(id)→Returns MailObj.
- change_mailbox(mailbox)→Changes mailbox
- quit→Close and Logout

Some Mail Objects that read the specific details from the mails are:

- title→ Returns a string of 'Subject' header
- sender→ Returns string of 'From' header.
- date→Returns string of 'Date' header.
- body→Returns string of Body.
- contenttype→Returns string of 'Content-Type' header.
- contenttransferencoding→Returns string of 'Content-Transfer-Encoding' header.
- references→Returns string of 'References' header.
- inreplyto→Returns string of 'In-Reply-To' header.
- replyto→Returns string of 'Reply-To' header.
- returnpath→Returns string of 'Return-Path' header.
- mimeversion→Returns string of 'MIME-Version' header.
- messageid→Returns string of 'Message-ID' header.
- attachments→Returns list of tuples ('attached file name', MailObj).

3.1.6 PSEUDO CODE

- Run the application by double-clicking on it.
- Then a voice command prompt saying to select one action Select the required action
 - If action is **“READ”**
 - It asks the serial number of mails which you want to read.

- Then it reads the contents in the mail
- If action is “**SEND**”
 - It asks for the mail id of the receiver
 - It asks to say the body of the mail
 - And then mail will be sent to that receiver.
- If action is “**EXIT**”
 - It Stops the Execution
- If the voice recognizer cannot able to recognize your voice, then prompts you to say again.
- This process is continuous until we give the EXIT command

3.2 IMPLEMENTATION

3.2.1 Execution Steps:

1. Install Python IDLE.
2. Install NLP Libraries.
 - SpeechRecognition and Pyaudio – to convert Speech to text
 - Pytsx3 – to convert text to Speech
 - Easyimap –to connect to Gmail server and to read/send emails.
3. When we run the application, A voice prompts and guides the user.
4. The Voice prompts operates based on the commands provided by the user.
5. For each command, there is a function associated with it.

3.2.1.1 Listen()

- Initially, we should create a recognizer class using **Recognizer()**
- When listen() is called, it takes the source using a microphone.

- Using **adjust_for_ambient_noise()** The recognizer class listens to the audio for the number of seconds specified from the beginning, then modifies the energy threshold value to make the entire audio more recognized.
- **recognize_google()** method on it to access the Google web speech API and turn spoken language into text. **recognize_google()** requires an argument **audio_data** otherwise it will return an error.

```
def listen():  
    with sr.Microphone() as source:  
        r.adjust_for_ambient_noise(source)  
        str1 = "speak Now"  
        speak(str1)  
        audio = r.listen(source)  
        try:  
            text = r.recognize_google(audio)  
            return text  
        except:  
            str1 = "Sorry could not recognize what have said"  
            speak(str1)
```

Figure:3.5 listen function

3.2.1.2. Speak()

- Initialize pytsx3 module using **init()** function.
- **Say()** is used to speak the text.
- We utilize **runAndWait()** to control the speech, Unless the interpreter encounters **runAndWait()**, none of the **say()** texts will be spoken.

```
def speak(str1):  
    print(str1)  
    engine.say(str1)  
    engine.runAndWait()
```

Figure:3.6 Speck function

3.2.1.3. readmail()

- This module reads the content in the mail
- To connect to IMAP server, first create an object specifying which server to connect to (in this case, the Gmail Server).

Server_obj = e.connect("imap.gmail.com",unm,pwd)

- **Server.listids()**, returns the IDs of the email in our unread inbox.
- **Email.date()** Returns string of 'Date' header.
- **Email.body()** Returns string of Body.
- **Email.title()** returns string of 'Subject' header.

```
def readmail():
    server = e.connect("imap.gmail.com",unm,pwd)
    server.listids()
    str1 = "please say the serial number of the email you wanna read starting from latest"
    speak(str1)
    a = listen()
    if(a == "Tu"):
        a = "2"
    b = int(a)-1
    email = server.mail(server.listids()[b])
    str1 = "The email is on: "
    speak(str1)
    speak(email.date)

    str1 = "The body of email is : "
    speak(str1)
    speak(email.body)
```

Figure 3.7 Readmail function

3.2.1.4. Sendmail()

- It calls listen() function to listen the details of the mail.
- `Server_obj= smtplib.SMTP_SSL("smtp.gmail.com",465)` is used to create SSL connection.
- To login we use `server.login("example@gmail.com", "password")`
- `server.sendmail(unm, rec, msg)` used to send the mail.
- `server.quit()` is used to quit the process of sending mail.

```
def sendmail():  
  
    str1 = "Please speak the body of your email"  
    speak(str1)  
    msg = listen()  
    str1 = "you have spoken the message"  
    speak(str1)  
    speak(msg)  
    server = smtplib.SMTP_SSL("smtp.gmail.com",465)  
    server.login(unm, pwd)  
    server.sendmail(unm , rec , msg)  
    server.quit()  
    str1 = "The mail has been sent"  
    speak(str1)
```

Figure 3.8 SendMail function

3.3 DATA PREPARATION

- A Voice guides the user to choose the specific service whether to read or send emails.
- It takes the user's voice as input.

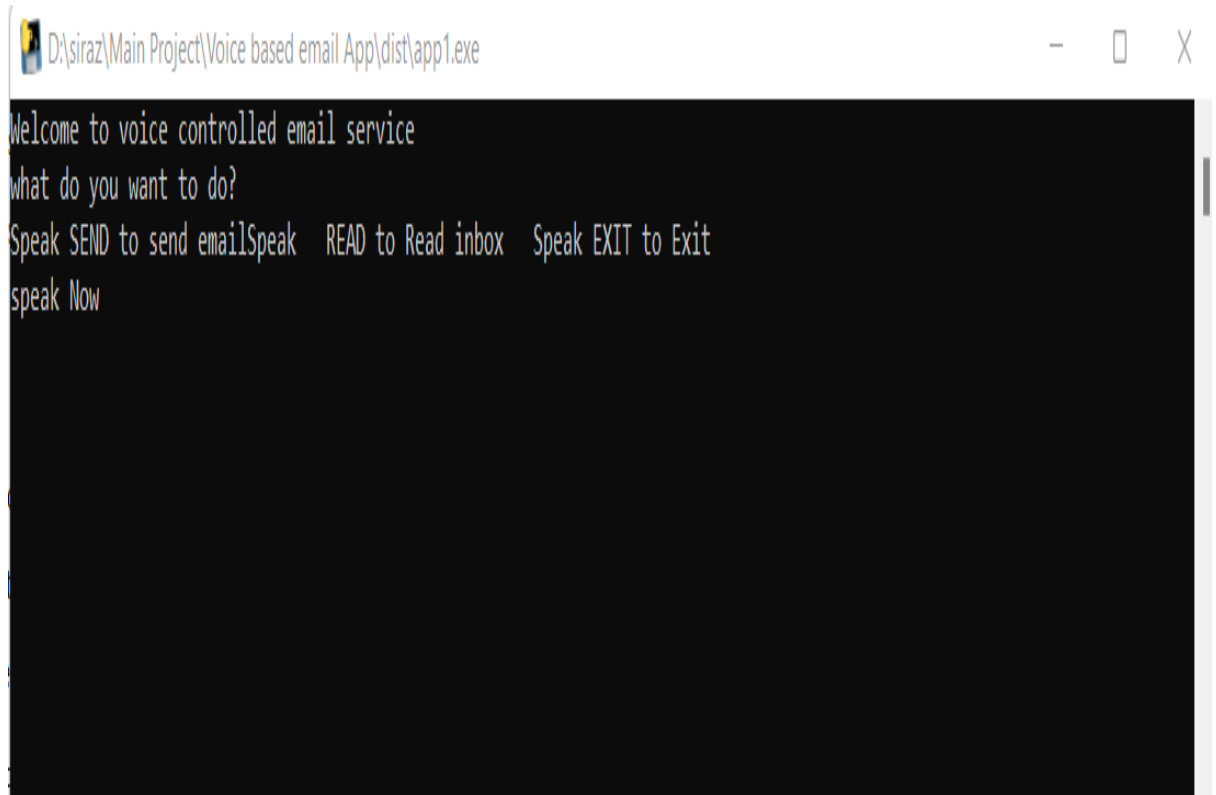


Figure 3.9 voice instructions to choose the service

CHAPTER – 4

RESULTS AND DISCUSSION

- Initially, a voice prompts to select the service whether the user wants to read or send email

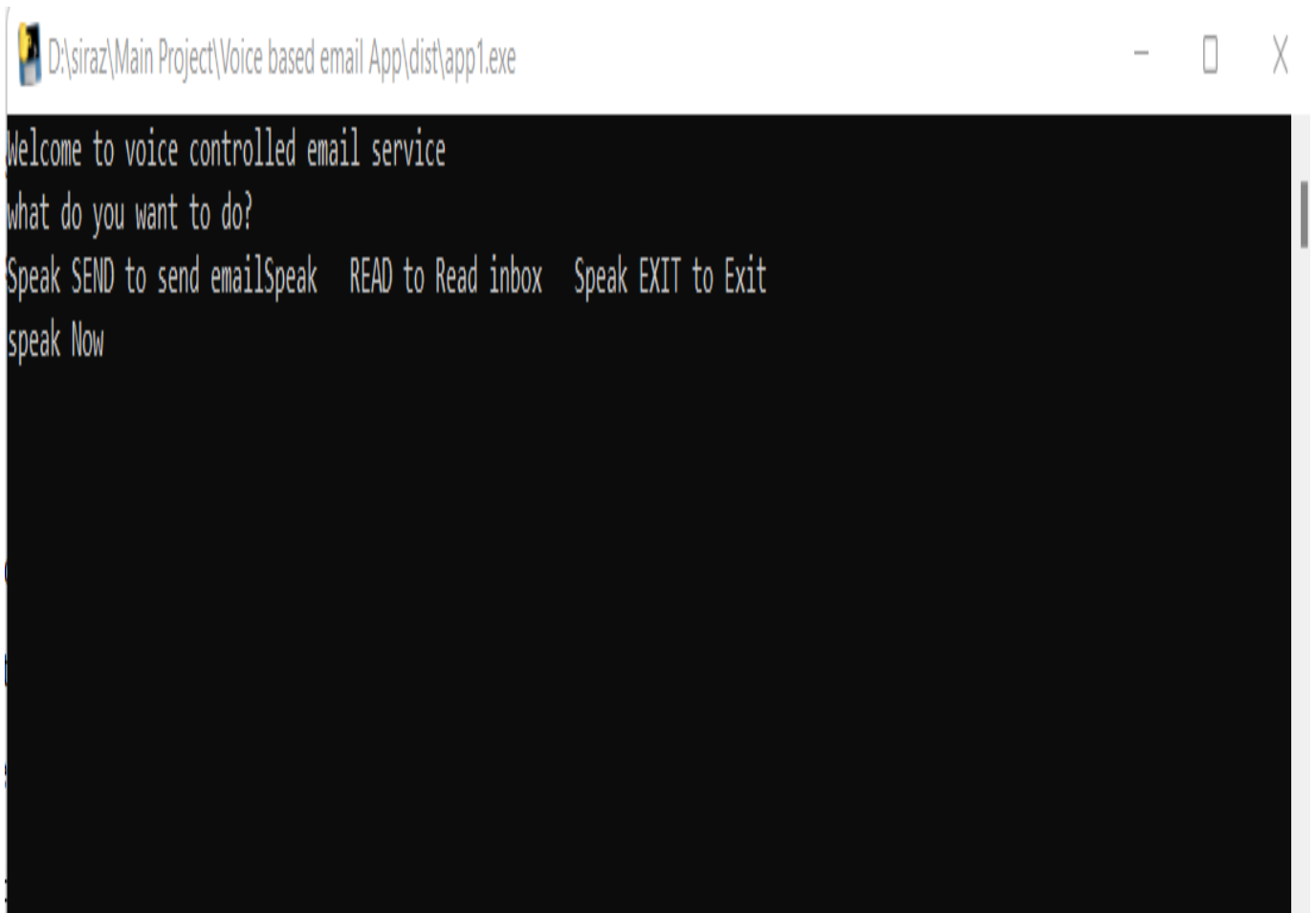


Figure 4.1 Initial Voice instruction

- If we select **the “SEND”** option then it asks to speak the body of the mail and the mail will be successfully send to the receiver as shown in the second image.

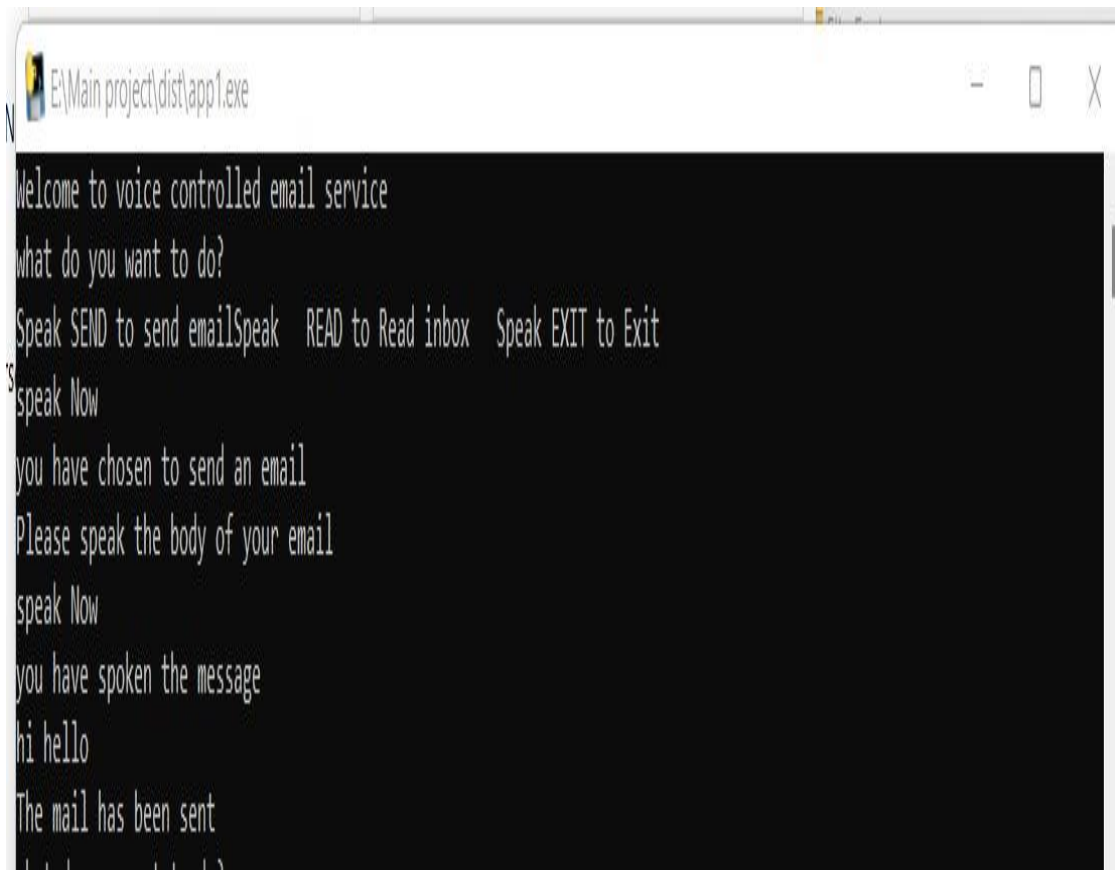


Figure : 4.2 When user chooses to send a mail

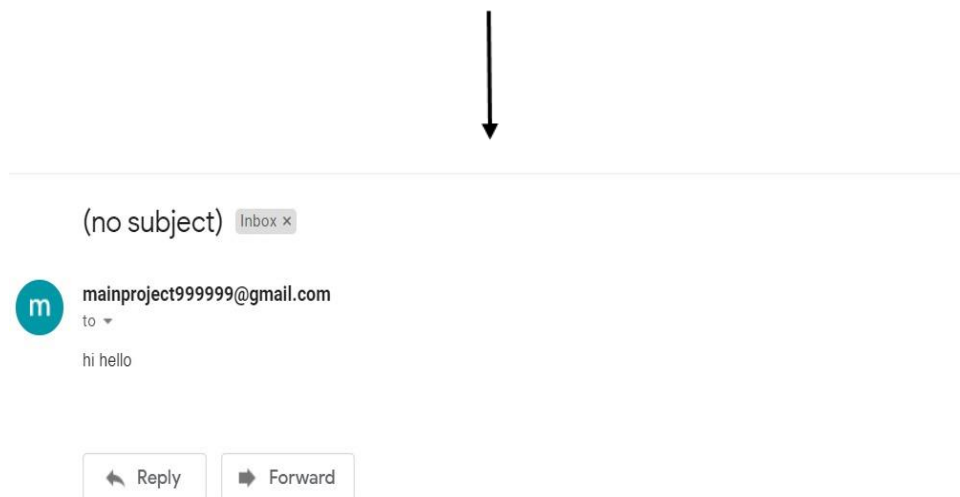
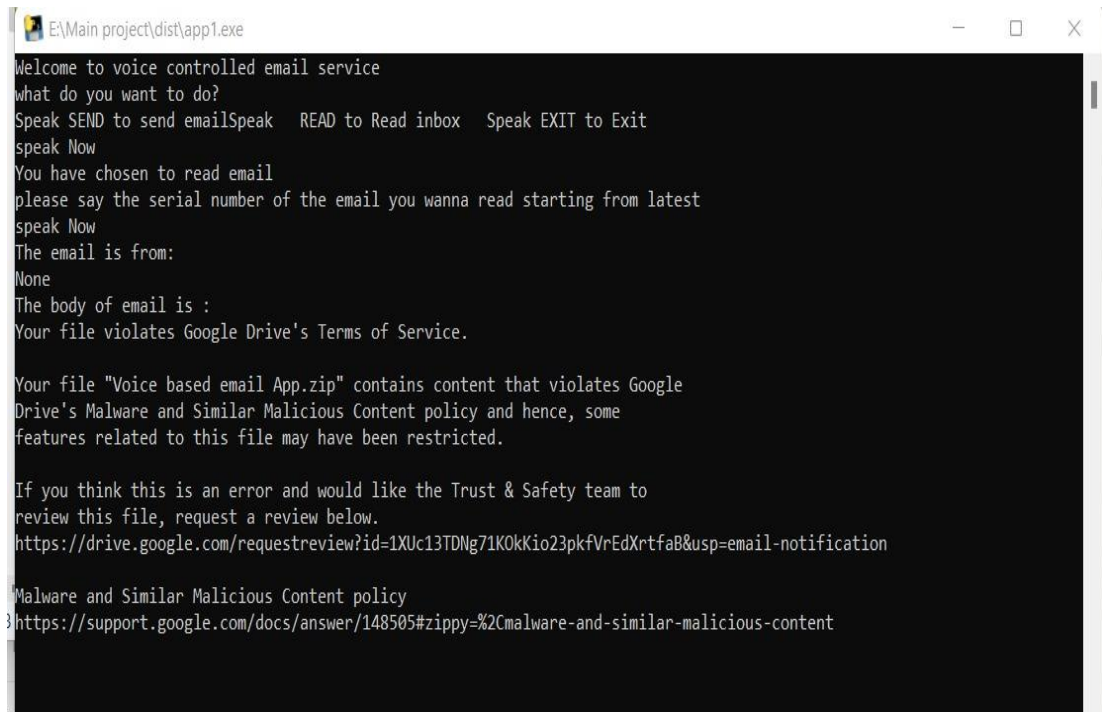


Figure 4.3 Mail send from voice is seen as send in normal gmail page

The first image shows that we can send the emails using the proposed system only using voice commands and the second image shows that mail has been successfully sent to the receiver.

- If we select the **"READ"** option



```
E:\Main project\dist\app1.exe
Welcome to voice controlled email service
what do you want to do?
Speak SEND to send emailSpeak READ to Read inbox Speak EXIT to Exit
speak Now
You have chosen to read email
please say the serial number of the email you wanna read starting from latest
speak Now
The email is from:
None
The body of email is :
Your file violates Google Drive's Terms of Service.

Your file "Voice based email App.zip" contains content that violates Google
Drive's Malware and Similar Malicious Content policy and hence, some
features related to this file may have been restricted.

If you think this is an error and would like the Trust & Safety team to
review this file, request a review below.
https://drive.google.com/requestreview?id=1XUc13TDNg71K0kKio23pkfVrEdXrtfaB&usp=email-notification

Malware and Similar Malicious Content policy
https://support.google.com/docs/answer/148505#zippy=%2Cmalware-and-similar-malicious-content
```

Figure 4.4 When user choose to read a email

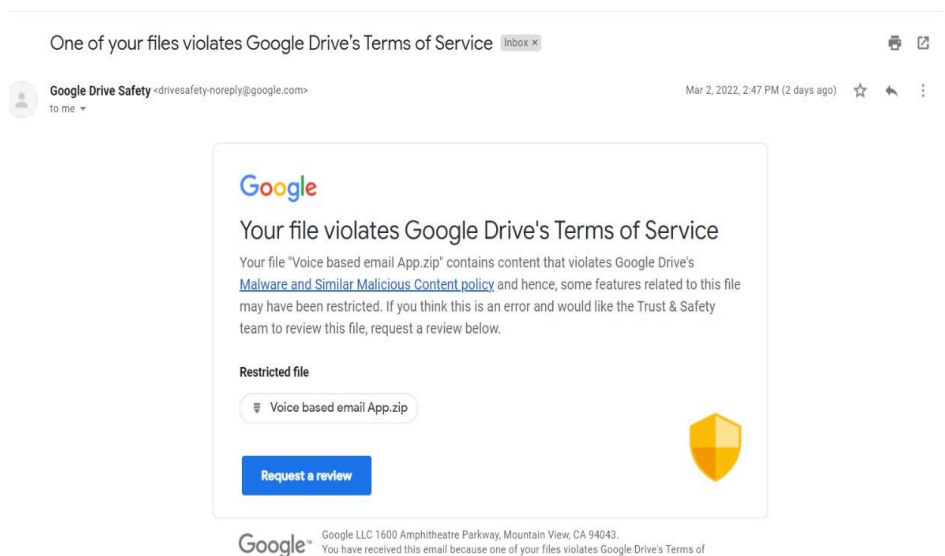


Figure 4.5 The mail which has been read using voice commands

In above images, it shows how the read operation has occurred. The voice prompt reads all the content in the mail when a user chooses the read operation.

- If we choose the “**EXIT**” option

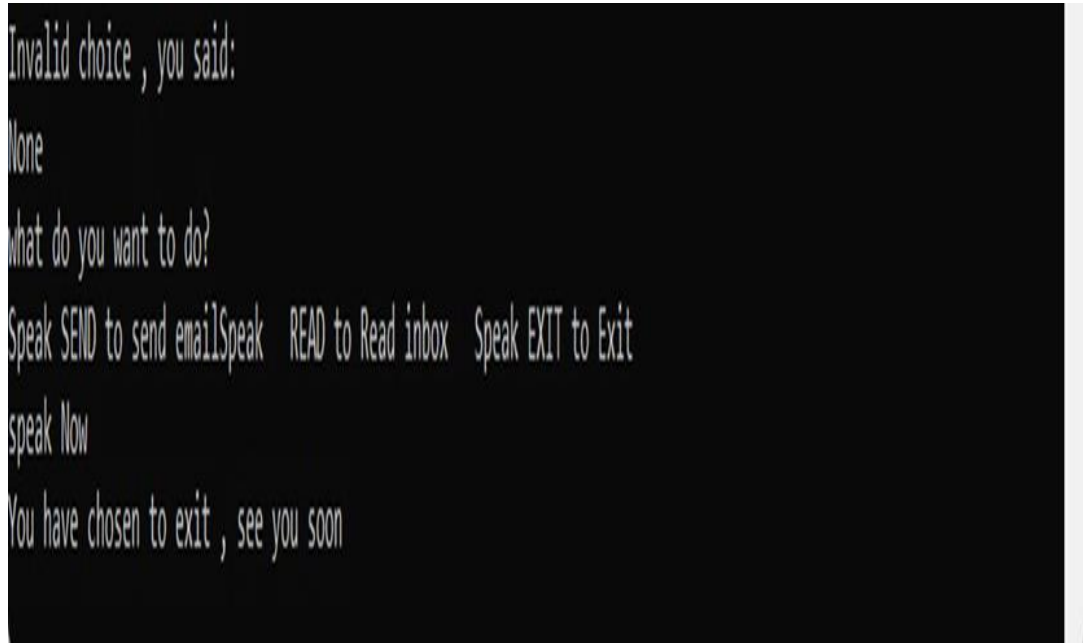


Figure 4.6 When user chooses to exit

When a user chooses the exit command then execution stops. And until you choose exit operation, the voice prompt moves in a continuous loop so that we can read and write any number of mails.

CHAPTER-5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

This project enables persons with visual impairments to participate in the development of digital India and to communicate more easily through the Internet and in people's lives. When you see how to send and receive an email, this technique removes many of the limitations that people have. The developers may be influenced by the project's success, motivating them to produce helpful items that can assist persons with low vision or who are blind.

5.2 FUTURE SCOPE

For people who can see, e-mailing is not a big deal, but for people who are not blessed with the gift of vision, it postures a key concern because of its intersection with many vocational responsibilities. So, this Voice-based email system helps them to use the email system. This system has a lot of potential in the future with a few improvements. This mechanism may also be enhanced to send an attachment, which is particularly useful for people who have a weak vision. It may be made available to anyone in the region and will remain available in a variety of languages. This System focuses more on the user-friendliness of all types of persons including regular persons, visually compromised people as well as illiterate.

BIBLIOGRAPHY

- [1] Mamatha, A., Jade, V., Saravana, J., Purshotham, A., & Suhas, A. V. (2020). Voice Based E-mail System for Visually Impaired. *International Journal of Research in Engineering, Science and Management*, 3(8), 51- 54.
- [2] Belekar, A., Sunka, S., Bhawar, N., & Bagade, S. Voice Based E-mail For The Visually Impaired. *International Journal of Computer Applications*, 975, 8887.
- [3] Khan, R., Sharma, P. K., Raj, S., Verma, S. K., & Katiyar, S. Voice Based E-Mail System using Artificial Intelligence.
- [4] Rijwan Khan,Pawan Kumar sharma,Sumit Raj,Sushil Kr. Verma, Sparsh Katiyar "Voice-Based E-Mail System using Artificial Intelligence".In *IJEAT* Volume 09, Issue 03,(February 2020)
- [5]. Payal Dudhbale J. S.Wankhade, P. S. Narawade . "Voice-Based System in Desktop and Mobile Devices for Blind People ". In *International Journal of Scientific Research in Science and Technology*, i2018.
- [6] Ruchi Khedekar, Sonu Gupta, i2019, Voice based email System for Blinds, *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT)* Volume i08, Issue i10 (October i2019).
- [7]. G. Shoba, G. Anusha, V.Jeevitha, R.Shanmathi."AN Interactive Email for Visually Impaired". In *International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE)*, i2014 ion Pages i5089-5092 (Volume i3, Issue i1).
- [8] G. Tejaswani, Afroz.B, Prof. Sunitha S (2016), "A Text Recognizing Device For Visually Impaired", in *International Journal of Engineering and Computer Science*, Vol.7, Issue.3, pp. 23697-23700
- [9] Pradeep Manohar, Aparajit Parthasarathy, "An Innovative Braille System Keyboard for the Visually Impaired".
- [10] K. Jayachandran, P. Anbumani (2017), "Voice Based Email for Blind People", in *International Journal of Advanced Research, Ideas and Innovations In Technology*, Vol.3, Issue.3,pp.1065-1071.

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions., component, or software to meet the desired needs.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able

to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

PSO1: Design, develop, test and maintain reliable software systems and intelligent systems.

PSO2: Design and develop web sites, web apps and mobile apps.

PROJECT PROFORMA

Classification of Project	Application	Product	Research	Review
	√			

Note: Tick Appropriate category

Project Outcomes	
Course Outcome (CO1)	Identify and analyze the problem statement using prior technical knowledge in the domain of interest.
Course Outcome (CO2)	Design and develop engineering solutions to complex problems by employing systematic approach.
Course Outcome (CO3)	Examine ethical, environmental, legal and security issues during project implementation.
Course Outcome (CO4)	Prepare and present technical reports by utilizing different visualization tools and evaluation metrics.

Mapping Table

CS1537: MAIN PROJECT															
Course Outcomes	Program Outcomes and Program Specific Outcome														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12		PSO 1	PSO 2
CO1	3	3	1					2	2	2				1	1
CO2	3	3	3	3	3			2	2	2		1		3	3
CO3	2	2	3	2	2	3	3	3	2	2	2			3	
CO4	2		1		3				3	3	2	2		2	2

Note: Map each project outcomes with POs and PSOs with either 1 or 2 or 3 based on level of mapping as follows:

1-Slightly (Low) mapped 2-Moderately (Medium) mapped 3-Substantially (High) mapped