

VOICE BASED MAIL ASSISTANT

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering
by

V TARUN ESWAR REDDY (Reg. No. 37110852)

Y SHANKAR TEJA (Reg. No. 37110862)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA



INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with “A” grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **V TARUN ESWAR REDDY (Reg. No. 37110852)** and **Y SHANKAR TEJA (Reg. No. 37110862)** who carried out the project entitled “**PROCTORING ONLINE EXAMS THROUGH MACHINE LEARNING**” under my supervision from **August 2020** to **March 2021**.

Internal Guide

Dr. R.M GOMATHI, M.TECH., PH.D.,

Head of the Department

Submitted for Viva voice Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I V TARUN ESWAR REDDY hereby declare that the Project Report entitled **“PROCTORING ONLINE EXAMS THROUGH MACHINE LEARNING”** is done by me under the guidance of **Dr. R.M GOMATHI M.TECH., PH.D.**, Department of Information Technology at Satyabhama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean, School of Computing, Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. R.M GOMATHI M.TECH., PH.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of **The Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Dynamic Online training is helping understudies and establishments worldwide to get to information base of wide assortment. This type of learning and training is expanding quickly, Evaluation and administering for the online courses has become a significant bottleneck for versatility of such learning frameworks. Manual human oversight is a typical methodology for test delegating and assessment where analyst needs to be present in the testing climate or necessities to screen testing climate of a test taker outwardly and acoustically through a webcam. In our proposed framework, we present a totally mechanized, test delegating arrangement that requires no human association. The framework coordinates every one of the contributions to measure and gauge the assortment of occasions, practices and examples regularly connected with cheating. By joining ceaseless character confirmation and programmed recognition of misbehavior or dubious exercises by an understudy, this framework gives a versatile, on the web, totally computerized, human connection free delegating framework that can be got to by test takers and executives to a really effective answer for traditional issue of online test administering.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	
	LIST OF FIGURES	
1	1.1 INTRODUCTION	1
	1.2 LITERATURE SURVEY	4
2.	AIM AND SCOPE	5
	2.1 AIM OF PROJECT	5
	2.2 OBJECTIVES	5
	2.3 SCOPE	5
3.	SYSTEM DESIGN & REQUIREMENTS	6
	3.1 EXISTING SYSTEM	6
	3.2 PROPOSED SYSTEM	6
	3.3 ADVANTAGES	7
	3.4 REQUIREMENT SPECIFICATION	7
	3.4.1 HARDWARE REQUIREMENTS	7
	3.4.2 SOFTWARE REQUIREMENTS	8
4.	SOFTWARE DEVELOPMENT METHODOLOGY	9
	4.1 PYTHON	9
	4.2 APPLICATIONS OF PYTHON	10
	4.3 FEATURES OF ANACONDA NAVIGATOR	11
	4.4 APPLICATIONS IN ANACONDA	12
	4.5 NEW FEATURES OF ANACONDA 5.3	13

CHAPTER NO	TITLE	PAGE NO
	4.6 LIBRARIES USED	14
	4.6.1 PYTTTSX3	14
	4.6.2 SPEECH RECOGNITION	17
	4.6.3 DATETIME LIBRARY	18
	4.6.4 WIKIPEDIA	18
	4.6.5 EMAIL	18
	4.6.6 PYWHATKIT	21
	4.6.7 SMTP PROTOCOL CLIENT	22
	4.7 INTERFACE SELECTION MODULE	32
	4.7.1 ARCHITECTURE PROPOSED	33
5.	RESULTS AND DISCUSSION	35
	5.1 RESULTS	35
6.	CONCLUSION AND FUTUREWORK	36
	6.1 CONCLUSION	36
	6.2 FUTURE WORK	36
	REFERENCES	37
	APPENDIX	
	A. SOURCE CODE	38
	B. SCREEN SHOTS	44
	C. PLAGIARISM REPORT	45

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The internet communication technology provides us the opportunity to keep in touch with the world through emails, video conferencing, instant messaging and more. But the blinds cannot use these facilities efficiently because all this requires visual perception. Internet is considered as a major storehouse of information in today's world. No single work can be done without the help of it. It has even become one of the de facto methods used in communication. And out of all methods available email is one of the most common forms of communication especially in the business world.

However not all people can use the internet. This is because in order to access the internet you would need to know what is written on the screen. If that is not visible it is of no use. This makes internet a completely useless technology for the visually impaired and illiterate people. Even the systems that are available currently like the screen readers TTS and ASR do not provide full efficiency to the blind people so as to use the internet. As nearly 285 million people worldwide are estimated visually impaired it become necessary to make internet facilities for communication usable for them also. Therefore, we have come up with this project in which we will be developing a voice-based email system which will aid the visually impaired people who are naive to computer systems to use email facilities in a hassle-free manner.

The users of this system would not need to have any basic information regarding keyboard shortcuts or where the keys are located. All functions are based on voice commands making it very easy for any type of user to use this system. Also, the user need not worry about remembering voice commands. he/she needs to perform in order to avail a given service as the system itself

will be prompting them as to which commands provide them with what operations.

A voice dependent email system is an android solution which allows the user to send and receive emails without using any visual features, which means everything can be done by voice commands. This application is designed and engineered especially for people with visual disabilities. According to a survey, around 250 million and more visually challenged or visually impaired are present in this world. That means around 250 million people are deprived or unaware of the science boon i.e., Internet.

They are fully dependent on the people with sight for their work. They are unable to use the internet and many facilities provided by it. Among that the main and basic feature is sending and reading mails. People who are visually impaired cannot access the normal type of email system so they have to depend on others, but this application will enable them to send and receive emails with voice commands and minimal action which can be carried out by themselves. So, by using this voice-based email system, they can take a step forward to independently use the benefits of the internet. This application is also useful for normal people who don't like to type so much, they can just use this application to compose a whole email by voice commands only. Project WE-Talk is a sort of aid to the visually impaired people by giving them access to the basic and mandatory features of the internet i.e., emails.

Internet is considered as a major storehouse of information in today's world. No single work can be done without the help of it. It has even become one of the de facto methods used in communication. And out of all methods available email is one of the most common forms of communication especially in the business world. However not all people can use the internet. This is because in order to access the internet you would need to know what is written on the screen. If that is not visible it is of no use. This makes internet a

completely useless technology for the visually impaired and illiterate people. Even the systems that are available currently like the screen readers TTS and ASR do not provide full efficiency to the blind people so as to use the internet. As nearly 285 million people worldwide are estimated visually impaired it become necessary to make internet facilities for communication usable for them also.

Therefore, we have come up with this project in which we will be developing a voice-based email system which will aid the visually impaired people who are naive to computer systems to use email facilities in a hassle-free manner. The users of this system would not need to have any basic information regarding keyboard shortcuts or where the keys are located. All functions are based on simple voice commands making it very easy for any type of user to use this system. Also, the user need not worry about remembering which mouse click operation he/she needs to perform in order to avail a given service as the system itself will be prompting them as to which click will provide them with what operations.

Voice-based personal assistant system has always seemed a little out of place in the enterprise. It is a useful tool for search, for reminders, and to write notes just by speaking it up. Intelligent Window assistant is used to create voice apps for intelligent assistant. When user needs to open any other application, he/she can use the command open. E.g., Open Notepad, File explorer, goggle chrome, this will help to open the applications.

When user wants to write some messages, he/she can use command write. And for searching purpose search command can be used. It will also restart and shutdown on the command. It will detect the speech and save in the database, and retrieve from the database and executive command and delete it from database. Interactions between a user and your Window assistant skills are mostly free-form, so assistant must understand language

naturally and also in context. Window assistant determines what a user wants to do by identifying the user intent from spoken or textual input by utterance. The intent maps utterances to actions that Window assistant can take, such as invoking a dialog.

1.2 LITERATURE SURVEY

Speech recognition has a long history with several waves of major innovations. Speech recognition for dictation, search, and voice commands has become a standard feature on smartphones and wearable devices. Voice-based personal assistant system has always seemed a little out of place in the enterprise. It is a useful tool for search, for reminders, and to write notes just by speaking it up. Intelligent Window assistant is used to create voice apps for intelligent assistant.

When user needs to open any other application, he/she can use the command open. E.g., Open Notepad, File explorer, goggle chrome, this will help to open the applications. When user wants to write some messages, he/she can use command write. And for searching purpose search command can be used. It will also restart and shutdown on the command. It will detect the speech and save in the database, and retrieve from the database and executive command and delete it from database. Interactions between a user and your Window assistant skills are mostly free-form, so assistant must understand language naturally and also in context. Window assistant determines what a user wants to do by identifying the user intent from spoken or textual input by utterance. The intent maps utterances to actions that Window assistant can take, such as invoking dialog.

CHAPTER 2

AIM AND SCOPE

2.1 AIM OF PROJECT

Voice based Mail system architecture is an application software which can be used by a Blind person to access Mails easily and efficiently. The developed application is interactive and it is user friendly and is efficient in accessing mail and has several extra features such as subject, BCC. The developed software also provides scope for several other operations like Wikipedia search, weather information. Accessing YouTube, google search engine. we can also shut down our pc through voice commands.

2.2 OBJECTIVES

Our application's main objective is to help the blind people to send and receive mails without any other person's help. The user only needs to speak the predefined keywords for each action to take place such as Login, Send, CC, BCC, Delete, etc. In this way it will become very easy to access mails for those who cannot see. So, the need to rely on other people for using mail will be reduced. The complete system is based on SIT (Speech to text), TTS (Text to Speech), IVR (interactive voice response). The application will detect the voice commands of user and responds accordingly.

2.3 SCOPE

When user needs to open any other application, he/she can use the command open. E.g., Open Notepad, File explorer, google chrome, this will help to open the applications. It will detect the speech and save in the database, and retrieve from the database and executive command and delete it from database. Interactions between a user and your Window assistant skills are mostly free-form, so assistant must understand language naturally and

also in context. Window assistant determines what a user wants to do by identifying the user.

CHAPTER 3

SYSTEM DESIGN & REQUIREMENTS

3.1 EXISTING SYSTEM

For a visually challenged person using a computer for the first time is not that convenient as it is for a normal user even though it is user friendly. Although there are many screen readers available then also these people face some minor difficulties. Screen readers read out whatever content is there on the screen and to perform those actions the person will have to use keyboard shortcuts as mouse location cannot be traced by the screen readers. This means two things; one that the user cannot make use of mouse pointer as it is completely inconvenient if the pointer location cannot be traced and second that user should be well versed with the keyboard as to where each and every key is located. A user is new to computer can therefore not use this service as they are not aware of the key locations.

This makes emails the most used form of communication. The most common mail services that we use in our day-to-day life cannot be used by visually challenged people. This is because they do not provide any facility so that the person in front can hear out the content of the screen. As they cannot visualize what is already present on screen they cannot make out where to click in order to perform the required operations. For a visually challenged person using a computer for the first time is not that convenient as it is for a normal user even though it is user friendly. Although there are many screen readers available then also these people face some minor difficulties.

3.2 PROPOSED SYSTEM

The proposed system is based on a completely novel idea and is nowhere like the existing mail systems. The most important aspect that has been kept in mind while developing the proposed system is accessibility. A web system is said to be perfectly accessible only if it can be used efficiently by all types of people whether able or disabled. The current systems do not provide this accessibility. Thus, the system we are developing is completely different from the current system. Unlike current system which emphasizes more on user friendliness of normal users, our system focuses more on user friendliness of all types of people including normal people visually impaired people as well as illiterate people.

When using this system, the computer will be prompting the user to perform specific operations to avail respective services and if the user needs to access the respective services, then he/she needs to perform that operation. One of the major advantages of this system is that user won't require to use the keyboard. All operations will be based on voice commands.

3.3 ADVANTAGES

- We can use the mail service with ease.
- Disabled people could operate hassle free.
- It is accurate and highly efficiency

3.4 REQUIREMENT SPECIFICATION

3.4.1 Hardware Requirements

The basic hardware required to run the program are:

- Hard disk of 20 GB

- System memory (RAM) of 4GB
- I3 processor-based computer or higher.

3.4.2 Software Requirements:

- Python
- SMTP Lib
- Tkinter

CHAPTER 4

SOFTWARE DEVELOPMENT METHODOLOGY

4.1 Python

Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc. Python programming is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability & this class will teach you python from basics.

Python Programming Characteristics

- It provides rich data types and easier to read syntax than any other programming languages
- It is a platform independent scripted language with full access to operating system API's
- Compared to other programming languages, it allows more run-time flexibility
- It includes the basic text manipulation facilities of Perl and Awk
- A module in Python may have one or more classes and free functions
- Libraries in Python are cross-platform compatible with Linux, Macintosh, and Windows
- For building large applications, Python can be compiled to byte-code

- Python supports functional and structured programming as well as OOP
- It supports interactive mode that allows interacting Testing and debugging of snippets of code
- In Python, since there is no compilation step, editing, debugging and testing is fast.

4.2 Applications of Python

Web Applications

You can create scalable Web Apps using frameworks and CMS (Content Management System) that are built on Python. Some of the popular platforms for creating Web Apps are: Django, Flask, Pyramid, Plone, Django CMS. Sites like Mozilla, Reddit, Instagram and PBS are written in Python.

Scientific and Numeric Computing

There are numerous libraries available in Python for scientific and numeric computing. There are libraries like: SciPy and NumPy that are used in general purpose computing. And, there are specific libraries like: EarthPy for earth science, AstroPy for Astronomy and so on. Also, the language is heavily used in machine learning, data mining and deep learning.

Creating software Prototypes

Python is slow compared to compiled languages like C++ and Java. It might not be a good choice if resources are limited and efficiency is a must. However, Python is a great language for creating prototypes. For example: You can use Pyramid (library for creating games) to create your game's prototype first. If you like the prototype, you can use language like C++ to create the actual game.

Good Language to Teach Programming

Python is used by many companies to teach programming to kids and newbies. It is a good language with a lot of features and capabilities. Yet, it's one of the easiest languages to learn because of its simple easy-to-use syntax.

4.3 FEATURES OF ANACONDA NAVIGATOR

Anaconda is a free and open source, easy to install distribution of Python and R programming languages. Anaconda provides a working environment which is used for scientific computing, data science, statistical analysis and machine learning.

The latest distribution of Anaconda is Anaconda 5.3 and is released in October, 2018. It has the conda package, environment manager and a collection of 1000+ open-source packages long with free community support.

Anaconda distribution and easily manage conda packages, environments and channels without the use of command-line commands. It is available for Windows, macOS and Linux.

4.4 Applications In Anaconda

The Anaconda distribution comes with the following applications along with Anaconda Navigator.

1. JupyterLab
2. Jupyter Notebook
3. Qt Console
4. Spyder
5. Glueviz
6. Orange3

7. RStudio

8. Visual Studio Code

> JupyterLab: This is an extensible working environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

> Jupyter Notebook: This is a web-based, interactive computing notebook environment. We can edit and run human-readable docs while describing the data analysis.

> Qt Console: It is the PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips and more.

> Spyder: Spyder is a scientific Python Development Environment. It is a powerful Python IDE with advanced editing, interactive testing, debugging and introspection features.

> VS Code: It is a streamlined code editor with support for development operations like debugging, task running and version control.

> Glue viz: This is used for multidimensional data visualization across files. It explores relationships within and among related datasets.

> Orange 3: It is a component-based data mining framework. This can be used for data visualization and data analysis. The workflows in Orange 3 are very interactive and provide a large toolbox.

> RStudio: It is a set of integrated tools designed to help you be more productive with R. It includes R essentials and notebooks.

4.5 New Features of Anaconda 5.3

Compiled with Latest Python release: Anaconda 5.3 is compiled with Python 3.7, taking advantage of Python's speed and feature improvements.

- **Better Reliability:** The reliability of Anaconda has been improved in the latest release by capturing and storing the package metadata for installed packages.
- **Enhanced CPU Performance:** The Intel Math Kernel Library 2019 for Deep Neural Networks (MKL 2019) has been introduced in Anaconda 5.3 distribution. Users deploying TensorFlow can make use of MKL 2019 for Deep Neural Networks. These Python binary packages are provided to achieve high CPU performance.
- **New packages are added:** There are over 230 packages which has been updated and added in the new release.
- **Work in Progress:** There is a casting bug in NumPy with Python 3.7 but the team is currently working on patching it until NumPy is updated.

4.6 LIBRARIES USED

4.6.1 PYTTX3

Voice computing is the discipline that develops hardware or software to process voice inputs. It spans many other fields including human-computer interaction, conversational computing, linguistics, natural language processing, automatic speech recognition, speech synthesis, audio engineering, digital signal processing, cloud computing, data science, ethics, law, and information security. Voice computing has become increasingly significant in modern times, especially with the advent of smart speakers like the Amazon Echo and Google Assistant, a shift towards serverless computing, and improved accuracy of speech recognition and text-to-speech models.

A voice computer is assembled hardware and software to process voice inputs. Note that voice computers do not necessarily need a screen, such as in the traditional Amazon Echo. In other embodiments, traditional laptop

computers or mobile phones could be used as voice computers. Moreover, there

has become increasingly more interfaces for voice computers with the advent of IoT-enabled devices, such as within cars or televisions. As of September 2018, there are currently over 20,000 types of devices compatible with Amazon Alexa. Software is Voice computing software can read/write, record, clean, encrypt/decrypt, playback, transcode, transcribe, compress, publish, futurize, model, and visualize voice files.

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3. An application invokes the `pyttsx3.init()` factory function to get a reference to a `pyttsx3`. Engine instance. it is a very easy to use tool which converts the entered text into Speech. The `pyttsx3` module supports two voices first is female and the second is male which is provided by “sapi5” for windows. Text-to-Speech (TTS) refers to the ability of computers to read text aloud. A TTS Engine converts written text to a phonemic representation, then converts the phonemic representation to waveforms that can be output as sound.

It supports three TTS engines: -

1. sapi5 – SAPI5 on Windows
2. nsss – NSSpeechSynthesizer on Mac OS X
3. espeak – eSpeak on every other platform

There are four sections in this tutorial:

1. Setup
2. Basic API
3. Results

Let's move on to the next section to find out more.

1. Setup

The proper way to install this module is by cloning the repository and installing via the setup Python file.

This is mainly because the published version in PyPI isn't the latest version at the time of this writing.

You can do a pip install in the future if the developer has published the latest version on PyPi.

EDIT: The package has been updated as of July 2020. You can install it easily via the following command.

```
->pip install pyttsx3
```

First, clone the repository from the official GitHub site, and unzip it to a directory of your choice.

Then, open up a command line, and activate the virtual environment you prefer. Once you're done, change the

directory to the root of the master folder. It should have a file called setup.py.

Run the following command:

```
python setup.py install
```

Once the installation has completed, kindly check if you have the following module:

```
pip show pyttsx3.
```

2. Basic API

In this section, we'll be exploring some of the useful functionalities provided in the module. If you

installed this module via the pip method, some of the features won't be available. Kindly refer to the

previous section to do a proper installation before you continue.

Let's start with a simple import statement

```
import pyttsx3
```

Initialization

The initialization is pretty straightforward — you can just type the following code:

```
engine = pyttsx3.init()
```

The engine object has the following variables that can be fine-tuned:

rate — Integer speech rate in words per minute. The base value is 200.

voice — String identifier of the active voice

volume — Floating point volume in the range of 0.0 to 1.0 inclusive

voices — List of pytsx3.voice. Voice descriptor objects

You can set them via the setProperty function.

```
engine.setProperty('rate', 150)
```

You can get the value of the property using the getProperty function.

```
engine.getProperty('volume')
```

Voice property

The voice property has the following variables:

age — Integer age of the voice in years. Defaults to None if unknown.

gender — String gender of the voice: male, female, or neutral. Defaults to None if unknown.

id — String identifier of the voice. Used to set the active voice via pytsx3.engine.Engine.setPropertyValue().

This attribute is always defined. Languages — List of string languages supported by this voice. Defaults to

an empty list of unknown.name — Human-readable name of the voice

4.6.2 SPEECH RECOGNITION SR

Speech recognition is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers. It is also known as automatic speech recognition (ASR), computer speech recognition or speech to text (STT). It incorporates knowledge and research in the computer science, linguistics and computer engineering fields. Some speech recognition systems require "training" (also called "enrolment") where an individual speaker reads text or isolated vocabulary into the system. The system analyses the person's specific voice and uses it to fine-tune the recognition of that person's speech, resulting in

increased accuracy. Systems that do not use training are called "speaker independent".

Speech recognition applications include voice user interfaces such as voice dialling (e.g. "call home"), call routing (e.g. "I would like to make a collect call"), domestic appliance control, search key words (e.g. find a podcast where particular words were spoken), simple data entry (e.g., entering a credit card number), preparation of structured documents (e.g. a radiology report), determining speaker characteristics, [speech-to-text processing (e.g., word processors or emails and aircraft (usually termed direct voice input). The term voice recognition or speaker identification refers to identifying the speaker, rather than what they are saying. Recognizing the speaker can simplify the task of translating speech in systems that have been trained on a specific person's voice or it can be used to authenticate or verify the identity of a speaker as part of a security process. From the technology perspective, speech recognition has a long history with several waves of major innovations. Most recently, the field has benefited from advances in deep learning and big data. The advances are evidenced not only by the surge of academic papers published in the field, but more importantly by the worldwide industry adoption of a variety of deep learning methods in designing and deploying speech recognition systems.

4.6.3 DATETIME LIB

In Python, date and time are not a data type of its own, but a module named datetime can be imported to work with the date as well as time. Datetime module comes built into Python, so there is no need to install it externally. Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps.

4.6.4 WIKIPEDIA

The Internet is the single largest source of information, and therefore it is important to know how to fetch data from various sources. And with Wikipedia being one of the largest and most popular sources for information on the Internet. Wikipedia is a multilingual online encyclopaedia created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system. In this we will see how to use Python's Wikipedia module to fetch a variety of information from the Wikipedia website.

Installation: -

In order to extract data from Wikipedia, we must first install the Python Wikipedia library, which wraps the official Wikipedia API. This can be done by entering the command below in your command prompt or terminal:

`pip install Wikipedia.`

Syntax: `Wikipedia. Summary (title, sentences)`

Argument:

Title of the topic

Optional argument: setting number of lines in result.

Return: Returns the summary in string format.

Voice search Wikipedia using Python: -

Every day, we visit so many applications, be it messaging platforms like Messenger, Telegram or ordering products on Amazon, Flipkart, or knowing about weather and the list can go on. And we see that these websites have their own software program for initiating conversations with human beings using rules or artificial intelligence. Users interact with the software program via a conversational interface through written or spoken text. This can be referred to as an assistant.

An assistant can be made using Natural Language Processing (NLP) one of the most promising fields of artificial intelligence that uses natural languages to enable human interactions with machines.

There are two main approaches to NLP:

rule-based methods: - That follow some pre specified rules and answer as per those rules.

statistical methods: - i.e., methods related to machine learning that learn on their own according to user inputs.

And we will learn about how to create your own assistant using statistical methods.

To create an assistant, we'll use the Python programming language, for it has all the modules required for building it. Secondly, Python is easy to understand, even if you don't have much experience with programming.

4.6.5 EMAIL

The email package is a library for managing email messages. It is specifically not designed to do any sending of email messages to SMTP (RFC 2821), NNTP, or other servers; those are functions of modules such as `smtplib` and `nnplib`. The email package attempts to be as RFC-compliant as possible, supporting RFC 5233 and RFC 6532, as well as such MIME-related RFCs as RFC 2045, RFC 2046, RFC 2047, RFC 2183, and RFC 2231.

The overall structure of the email package can be divided into three major components, plus a fourth component that controls the behavior of the other components.

The central component of the package is an "object model" that represents email messages. An application interacts with the package primarily through the object model interface defined in the message sub-module. The application can use this API to ask questions about an existing email, to construct a new email, or to add or remove email subcomponents that themselves use the same object model interface. That is,

following the nature of email messages and their MIME subcomponents, the email object model is a tree structure of objects that all provide the `EmailMessage` API.

The other two major components of the package are the parser and the generator. The parser takes the serialized version of an email message (a stream of bytes) and converts it into a tree of `EmailMessage` objects. The generator takes an `EmailMessage` and turns it back into a serialized byte stream. (The parser and generator also handle streams of text characters, but this usage is discouraged as it is too easy to end up with messages that are not valid in one way or another.)

The control component is the policy module. Every `EmailMessage`, every generator, and every parser has an associated policy object that controls its behavior. Usually an application only needs to specify the policy when an `EmailMessage` is created, either by directly instantiating an `EmailMessage` to create a new email, or by parsing an input stream using a parser. But the policy can be changed when the message is serialized using a generator. This allows, for example, a generic email message to be parsed from disk, but to serialize it using standard SMTP settings when sending it to an email server.

The email package does its best to hide the details of the various governing RFCs from the application. Conceptually the application should be able to treat the email message as a structured tree of unicode text and binary attachments, without having to worry about how these are represented when serialized. In practice, however, it is often necessary to be aware of at least some of the rules governing MIME messages and their structure, specifically the names and nature of the MIME “content types” and how they identify multipart documents. For the most part this knowledge should only be required for more complex applications, and even then it should only be the high level structure in question, and not the details of how those structures are

represented. Since MIME content types are used widely in modern internet software (not just email), this will be a familiar concept to many programmers.

The following sections describe the functionality of the email package. We start with the message object model, which is the primary interface an application will use, and follow that with the parser and generator components. Then we cover the policy controls, which completes the treatment of the main components of the library.

The next three sections cover the exceptions the package may raise and the defects (non-compliance with the RFCs) that the parser may detect. Then we cover the headerregistry and the contentmanager sub-components, which provide tools for doing more detailed manipulation of headers and payloads, respectively. Both of these components contain features relevant to consuming and producing non-trivial messages, but also document their extensibility APIs, which will be of interest to advanced applications.

Following those is a set of examples of using the fundamental parts of the APIs covered in the preceding sections.

The foregoing represent the modern (unicode friendly) API of the email package. The remaining sections, starting with the Message class, cover the legacy compat32 API that deals much more directly with the details of how email messages are represented. The compat32 API does not hide the details of the RFCs from the application, but for applications that need to operate at that level, they can be useful tools. This documentation is also relevant for applications that are still using the compat32 API for backward compatibility reasons.

4.6.6 PYWHATKIT

Python offers numerous inbuilt libraries to ease our work. Among them pywhatkit is a Python library for sending WhatsApp messages at a certain time, it

has several other features too.

Following are some features of pywhatkit module:

1. Send WhatsApp messages.
2. Play a YouTube video.
3. Perform a Google Search.
4. Get information on particular topic.

In Python3 pywhatkit module will not come pre-installed, so you can install it by using the command:

```
pip install pywhatkit
```

1. Send Whatsapp Messages:

Here, we will learn the simplest way of using pywhatkit module which utilises the WhatsApp webpage to automate messages sending to any number on WhatsApp. But make sure that you have logged into your WhatsApp in your browser.

2. Play a YouTube video:

Function `pywhatkit.playonyt()`, opens the YouTube in your default browser and plays the video you mentioned in the function. If you pass the topic name as parameter, it plays the random video on that topic. On passing the URL of the video as the parameter, it open that exact video.

Syntax: `pywhatkit.playonyt("url/topic name")`

Parameters:

- URL/Topic Name: Mention the topic name or URL of the YouTube video

3. Perform Google Search:

You can perform a Google search using the following simple command. It

opens your browser and searches for the topic you have given in your code.

Syntax: `pywhatkit.search("Keyword")`

Parameters:

- Keyword: It open your browser and performs search operation.

4. Get information on particular topic:

We can get brief information on a particular topic. We can also limit the number of lines to be printed. Also, make sure that you are searching for the topics that are available on Wikipedia.

Syntax: `pywhatkit.info("topic",lines=x)`

Parameters:

- Topic: Gives the output on the topic mentioned.
- lines: It prints the searched topic in the specified number of lines.

4.6.7 SMTP PROTOCOL CLIENT

The `smtplib` module defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. For details of SMTP and ESMTP operation, consult RFC 821 (Simple Mail Transfer Protocol) and RFC 1869 (SMTP Service Extensions).

```
class smtplib.SMTP(host="", port=0, local_hostname=None, [timeout,  
]source_address=None)
```

An SMTP instance encapsulates an SMTP connection. It has methods that support a full repertoire of SMTP and ESMTP operations. If the optional host and port parameters are given, the `SMTP connect()` method is called with those

parameters during initialization. If specified, `local_hostname` is used as the FQDN of the local host in the HELO/EHLO command. Otherwise, the local hostname is found using `socket.getfqdn()`. If the `connect()` call returns anything other than a success code, an `SMTPConnectError` is raised. The optional timeout parameter specifies a timeout in seconds for blocking operations like the connection attempt (if not specified, the global default timeout setting will be used). If the timeout expires, `socket.timeout` is raised. The optional `source_address` parameter allows binding to some specific source address in a machine with multiple network interfaces, and/or to some specific source TCP port. It takes a 2-tuple (host, port), for the socket to bind to as its source address before connecting. If omitted (or if host or port are "" and/or 0 respectively) the OS default behaviour will be used.

For normal use, you should only require the initialization/connect, `sendmail()`, and `SMTP.quit()` methods. An example is included below.

The SMTP class supports the `with` statement. When used like this, the SMTP QUIT command is issued automatically when the `with` statement exits.

All commands will raise an auditing event `smtplib.SMTP.send` with arguments `self` and `data`, where `data` is the bytes about to be sent to the remote host.

Changed in version 3.9: If the timeout parameter is set to be zero, it will raise a `ValueError` to prevent the creation of a non-blocking socket

```
class smtplib.SMTP_SSL(host="", port=0, local_hostname=None, keyfile=None,
certfile=None, [timeout, ]context=None, source_address=None)
```

An `SMTP_SSL` instance behaves exactly the same as instances of `SMTP`. `SMTP_SSL` should be used for situations where SSL is required from the beginning of the connection and using `starttls()` is not appropriate. If `host` is not specified, the local host is used. If `port` is zero, the standard SMTP-over-SSL port (465) is arguments `local_hostname`, `timeout` and `source_address` have the same meaning as they do in the `SMTP` class. `context`, also optional, can contain a

SSLContext and allows configuring various aspects of the secure connection. Please read Security considerations for best practices.

keyfile and certfile are a legacy alternative to context, and can point to a PEM formatted private key and certificate chain file for the SSL connection.

Changed in version 3.3: context was added.

Changed in version 3.3: source_address argument was added.

Changed in version 3.4: The class now supports hostname check with ssl.SSLContext.check_hostname and ServerName Indication (see ssl.HAS_SNI).

Deprecated since version 3.6: keyfile and certfile are deprecated in favor of context. Please use ssl.SSLContext.load_cert_chain() instead, or let ssl.create_default_context() select the system's trusted CA certificates for you.

Changed in version 3.9: If the timeout parameter is set to be zero, it will raise a ValueError to prevent the creation of a non-blocking socket

```
class smtplib.LMTP(host="", port=LMTP_PORT, local_hostname=None,
source_address=None[, timeout])
```

The LMTP protocol, which is very similar to ESMTP, is heavily based on the standard SMTP client. It's common to use Unix sockets for LMTP, so our connect() method must support that as well as a regular host:port server. The optional arguments local_hostname and source_address have the same meaning as they do in the SMTP class. To specify a Unix socket, you must use an absolute path for host, starting with a '/'.

Authentication is supported, using the regular SMTP mechanism. When using a Unix socket, LMTP generally don't support or require any authentication, but your mileage might vary.

Changed in version 3.9: The optional timeout parameter was added.

A nice selection of exceptions is defined as well:

exception smtplib.SMTPException

Subclass of OSError that is the base exception class for all the other exceptions provided by this module.

Changed in version 3.4: SMTPException became subclass of OSError

exception smtplib.SMTPServerDisconnected

This exception is raised when the server unexpectedly disconnects, or when an attempt is made to use the SMTP instance before connecting it to a server.

exception smtplib.SMTPResponseException

Base class for all exceptions that include an SMTP error code. These exceptions are generated in some instances when the SMTP server returns an error code. The error code is stored in the smtp_code attribute of the error, and the smtp_error attribute is set to the error message.

exception smtplib.SMTPSenderRefused

Sender address refused. In addition to the attributes set by on all SMTPResponseException exceptions, this sets 'sender' to the string that the SMTP server refused.

exception smtplib.SMTPRecipientsRefused

All recipient addresses refused. The errors for each recipient are accessible through the attribute recipients, which is a dictionary of exactly the same sort as SMTP.sendmail() returns.

exception smtplib.SMTPDataError

The SMTP server refused to accept the message data.

exception smtplib.SMTPConnectError

Error occurred during establishment of a connection with the server.

exception smtplib.SMTPHeloError

The server refused our HELO message.

exception smtplib.SMTPNotSupportedError

The command or option attempted is not supported by the server.

exception smtplib.SMTPAuthenticationError

SMTP authentication went wrong. Most probably the server didn't accept the username/password combination provided.

SMTP.login(user, password, *, initial_response_ok=True)

Log in on an SMTP server that requires authentication. The arguments are the username and the password to authenticate with. If there has been no previous EHLO or HELO command this session, this method tries ESMTP EHLO first. This method will return normally if the authentication was successful, or may raise the following exceptions:

SMTPHeloError

The server didn't reply properly to the HELO greeting.

SMTPAuthenticationError

The server didn't accept the username/password combination.

SMTPNotSupportedError

The AUTH command is not supported by the server.

SMTPException

No suitable authentication method was found.

Each of the authentication methods supported by smtplib are tried in turn if they are advertised as supported by the server. See `auth()` for a list of supported authentication methods. `initial_response_ok` is passed through to `auth()`.

Optional keyword argument `initial_response_ok` specifies whether, for authentication methods that support it, an “initial response” as specified in RFC 4954 can be sent along with the AUTH command, rather than requiring a challenge/response.

Changed in version 3.5: `SMTPNotSupportedError` may be raised, and the `initial_response_ok` parameter was added.

```
SMTP.auth(mechanism, authobject, *, initial_response_ok=True)
```

Issue an SMTP AUTH command for the specified authentication mechanism, and handle the challenge response via `authobject`.

`mechanism` specifies which authentication mechanism is to be used as argument to the AUTH command; the valid values are those listed in the `auth` element of `esmtplib.features`.

`authobject` must be a callable object taking an optional single argument:

```
data = authobject(challenge=None)
```

If optional keyword argument `initial_response_ok` is true, `authobject()` will be called first with no argument. It can return the RFC 4954 “initial response” ASCII str which will be encoded and sent with the AUTH command as below. If the `authobject()` does not support an initial response (e.g. because it requires a challenge), it should return `None` when called with `challenge=None`. If `initial_response_ok` is false, then `authobject()` will not be called first with `None`.

If the initial response check returns `None`, or if `initial_response_ok` is false,

authobject() will be called to process the server's challenge response; the challenge argument it is passed will be a bytes. It should return ASCII str data that will be base64 encoded and sent to the server.

User code does not normally need to call auth directly, but can instead call the login() method, which will try each of the above mechanisms in turn, in the order listed. auth is exposed to facilitate the implementation of authentication methods not (or not yet) supported directly by smtplib.

New in version 3.5.

SMTP.starttls(keyfile=None, certfile=None, context=None)

Put the SMTP connection in TLS (Transport Layer Security) mode. All SMTP commands that follow will be encrypted. You should then call ehlo() again.

If keyfile and certfile are provided, they are used to create an ssl.SSLContext.

Optional context parameter is an ssl.SSLContext object; This is an alternative to using a keyfile and a certfile and if specified both keyfile and certfile should be None.

If there has been no previous EHLO or HELO command this session, this method tries ESMTP EHLO first.

SMTPHeloError

The server didn't reply properly to the HELO greeting.

SMTPNotSupportedError

The server does not support the STARTTLS extension.

RuntimeError

SSL/TLS support is not available to your Python interpreter.

Changed in version 3.3: context was added.

Changed in version 3.4: The method now supports hostname check with `SSLContext.check_hostname` and Server Name Indicator (see `HAS_SNI`).

Changed in version 3.5: The error raised for lack of STARTTLS support is now the `SMTPNotSupportedError` subclass instead of the base `SMTPException`.

`SMTP.sendmail(from_addr, to_addrs, msg, mail_options=(), rcpt_options=())`

Send mail. The required arguments are an RFC 822 from-address string, a list of RFC 822 to-address strings (a bare string will be treated as a list with 1 address), and a message string. The caller may pass a list of ESMTP options (such as `8bitmime`) to be used in MAIL FROM commands as `mail_options`. ESMTP options (such as DSN commands) that should be used with all RCPT commands can be passed as `rcpt_options`. (If you need to use different ESMTP options to different recipients you have to use the low-level methods such as `mail()`, `rcpt()` and `data()` to send the message.

Note

The `from_addr` and `to_addrs` parameters are used to construct the message envelope used by the transport agents. `sendmail` does not modify the message headers in any way.

`msg` may be a string containing characters in the ASCII range, or a byte string. A string is encoded to bytes using the `ascii` codec, and lone `\r` and `\n` characters are converted to `\r\n` characters. A byte string is not modified.

If there has been no previous EHLO or HELO command this session, this method tries ESMTP EHLO first. If the server does ESMTP, message size and each of the specified options will be passed to it (if the option is in the feature set the server advertises). If EHLO fails, HELO will be tried and ESMTP options suppressed.

This method will return normally if the mail is accepted for at least one recipient. Otherwise it will raise an exception. That is, if this method does not raise an exception, then someone should get your mail. If this method does not raise an exception, it returns a dictionary, with one entry for each recipient that was refused. Each entry contains a tuple of the SMTP error code and the accompanying error message sent by the server.

If SMTPUTF8 is included in mail_options, and the server supports it, from_addr and to_addrs may contain non-ASCII characters.

This method may raise the following exceptions:

SMTPRecipientsRefused

All recipients were refused. Nobody got the mail. The recipients attribute of the exception object is a dictionary with information about the refused recipients (like the one returned when at least one recipient was accepted).

SMTPHeloError

The server didn't reply properly to the HELO greeting.

SMTPSenderRefused

The server didn't accept the from_addr.

SMTPDataError

The server replied with an unexpected error code (other than a refusal of a recipient).

SMTPNotSupportedError

SMTPUTF8 was given in the mail_options but is not supported by the server.

Unless otherwise noted, the connection will be open even after an exception is

raised.

Changed in version 3.2: msg may be a byte string.

```
SMTP.send_message(msg, from_addr=None, to_addrs=None, mail_options=(),  
rcpt_options=())
```

This is a convenience method for calling `sendmail()` with the message represented by an `email.message.Message` object. The arguments have the same meaning as for `sendmail()`, except that `msg` is a `Message` object.

If `from_addr` is `None` or `to_addrs` is `None`, `send_message` fills those arguments with addresses extracted from the headers of `msg` as specified in RFC 5322: `from_addr` is set to the `Sender` field if it is present, and otherwise to the `From` field. `to_addrs` combines the values (if any) of the `To`, `Cc`, and `Bcc` fields from `msg`. If exactly one set of `Resent-*` headers appear in the message, the regular headers are ignored and the `Resent-*` headers are used instead. If the message contains more than one set of `Resent-*` headers, a `ValueError` is raised, since there is no way to unambiguously detect the most recent set of `Resent-` headers.

`send_message` serializes `msg` using `BytesGenerator` with `\r\n` as the `linesep`, and calls `sendmail()` to transmit the resulting message. Regardless of the values of `from_addr` and `to_addrs`, `send_message` does not transmit any `Bcc` or `Resent-Bcc` headers that may appear in `msg`. If any of the addresses in `from_addr` and `to_addrs` contain non-ASCII characters and the server does not advertise `SMTPUTF8` support, an `SMTPNotSupported` error is raised. Otherwise the `Message` is serialized with a clone of its policy with the `utf8` attribute set to `True`, and `SMTPUTF8` and `BODY=8BITMIME` are added to `mail_options`.

New in version 3.5: Support for internationalized addresses (`SMTPUTF8`).

```
SMTP.quit()
```

Terminate the SMTP session and close the connection. Return the result of the SMTP QUIT command.

Low-level methods corresponding to the standard SMTP/ESMTP commands HELP, RSET, NOOP, MAIL, RCPT, and DATA are also supported. Normally these do not need to be called directly, so they are not documented here.

4.7 Interface Selection Module

The Interface selection module provides an option to mention the type of user using the system. There are two options to be chosen, a) Blind user and b) sighted user. The difference between the two types of user modules is that, for Blind users, all operation performed will get a voice-based feedback on the other hand, a sighted person will receive textual feedbacks. Moreover, there are options to save a particular users profile so that the user does not have to enter the same details again.

Mailing Options

The mailing option has got the following two modules a) Compose Mail and b) Inbox Check. A user can choose any one option depending upon the task in hand.

Compose Mail

within the compose mail module, the compose window can open up, giving the user the choice either to A text or record a voice message. so as to record a voice message a user will use the beginning record command. so as to prevent the recording, once more the user use the stop recording command. Once the recording is over, the system can raise the user to pick the recipients address. Once the recipient mail id is entered, the system can prompt the user to send the mail or to cancel the operation. By default the user is within the home page if the user desires to maneuver to the compose page the command compose compose. In order to send the mail the command send is employed.

4.7.1 ARCHITECTURE PROPOSED

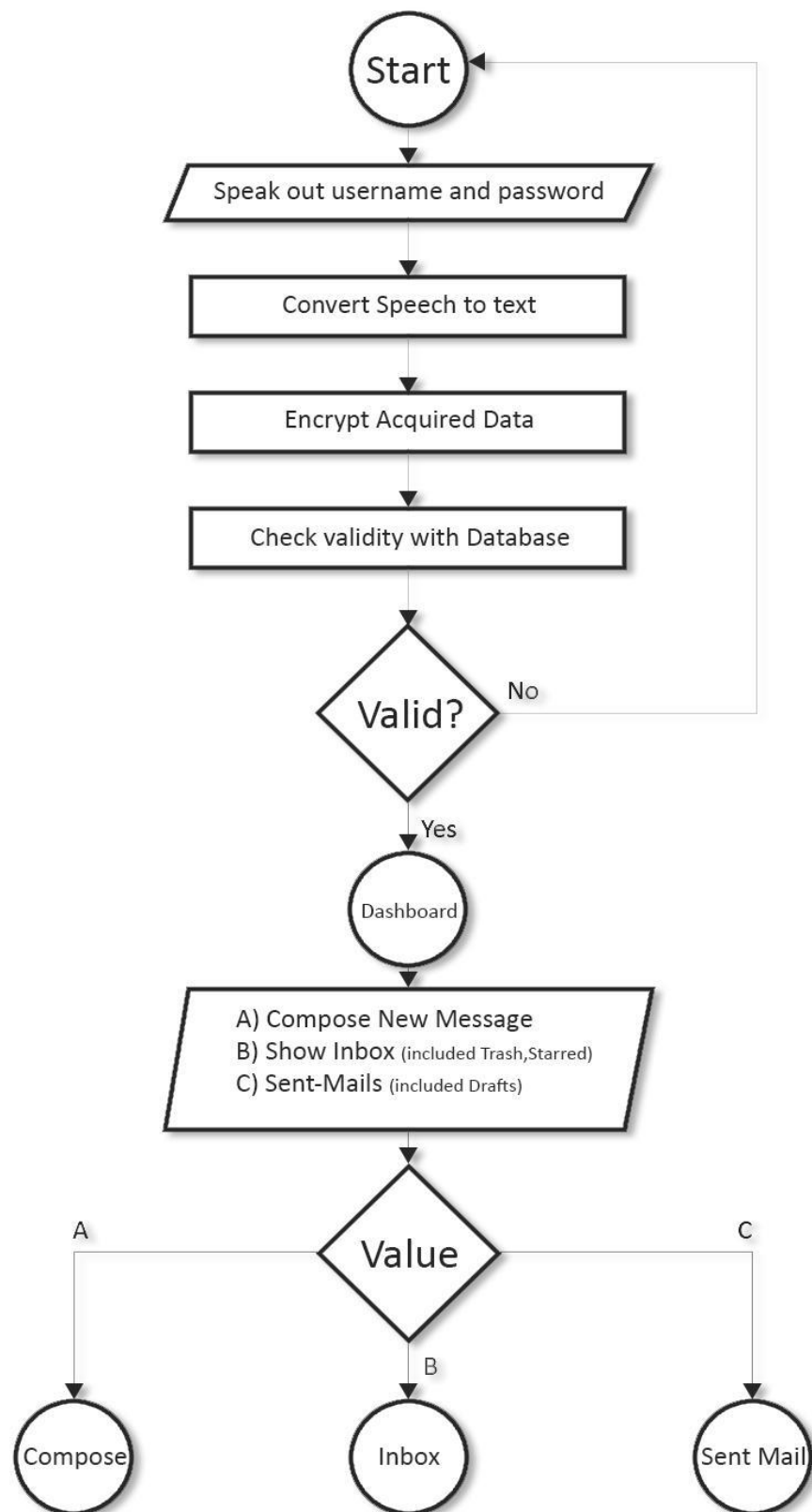


Fig. 4.1 Block Diagram of the VoiceMail Architecture

The algorithm works in the way that the system detects the most appropriate word when user spells it and then matches the word that is guessed with the actual word that is pronounced by the user. If both of them is found to be same then the word is selected from the dictionary and typed by the system without giving any burden to the user. Similarly the entire text is typed while the user simply spells it by the system itself so that the blind user need not use the keyboard or mouse for this purpose. That is how the algorithm is implemented to convert speech to text and text to speech.

The proposed system is a desktop application that allows sending and receiving of mails via the internet. We use artificial intelligence to benefit the blind to make use of the advanced technology for their growth and improvement. The proposed system is a desktop application which makes use of artificial intelligence that makes it cost-effective and easy to maintain. The proposed system makes use of the Viterbi algorithm for voice detection and conversion. It overcomes the disadvantages of the existing system in that it is fully voice based and gives no work to the mouse or keyboard. Since it uses voices, it provides an intuitive, as well as interactive and easy to use GUI that can be used by a blind user even if they are not computer literate. The idea is to create a class in .Net with .Net SMTP option.

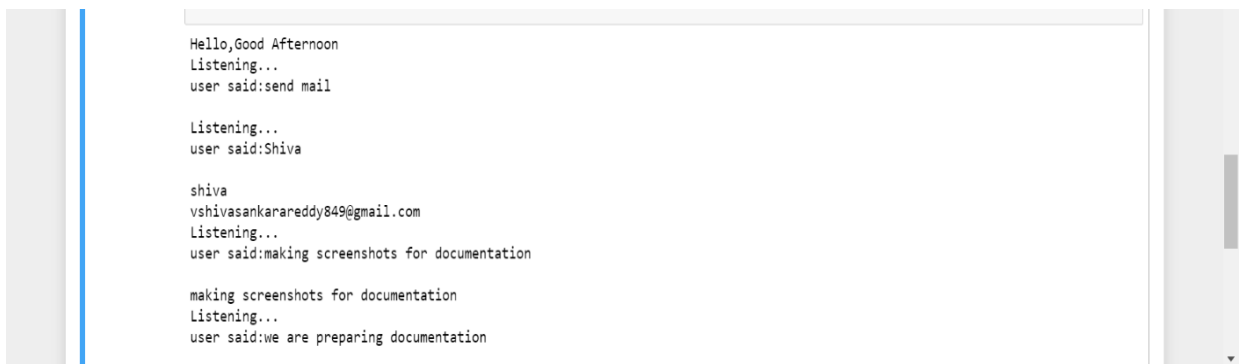
Voice Authentication is enabled using a simple mfcc features. The dictation mode is activated and events are generated when the user tells the number respective to the commands like send compose and read. The speech is typed into text in text box editor. When the number corresponding to “send” is said the mail is sent and when number corresponding to “read” is said the mail is read.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Results

This mail system based on voice is very effective and helpful for people who are visually impaired. This system also has the ability to convert speech to text and text to speech with the help of speech reader which is useful for blind people as well as visually impaired person. The application will detect the voice commands of user and with the help of interactive voice response user can interact with the application easily and efficiently. The user only needs to speak the predefined keywords for each action to take place such as Login, Send, CC, BCC, Delete, etc. To make the application more secure encryption and decryption algorithms can be used to increase the users trust while logging in. The algorithm can be made smart.



```
Hello,Good Afternoon
Listening...
user said:send mail

Listening...
user said:Shiva

shiva
vshivasankarareddy849@gmail.com
Listening...
user said:making screenshots for documentation

making screenshots for documentation
Listening...
user said:we are preparing documentation
```

FIGURE 5.1 Output Screen

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The proposed a system which will help the visually impaired people to access email services efficiently. This system will help in overcoming some drawbacks that were earlier faced by the blind people in accessing emails. We have eliminated the concept of using keyboard shortcuts along with screen readers which will help reducing the cognitive load of remembering keyboard shortcuts. Also, any naive user who does not know the location of keys on the keyboard need not worry as keyboard usage is eliminated. The user only needs to follow the instructions given by the IVR and use mouse clicks accordingly to get the respective services offered. Other than this the user might need to feed in information through voice inputs when specified.

6.2 FUTURE WORK

For the further development of the application, the attachments like images, word documents, audio and video files can be incorporated. Encryption and decryption algorithm can used to protect the username and password that is passed during login. More commands can be used to for different operations like search, mark important, delete, archive, go back, report spam, forward. Automated replying to received mails can be also integrated. The application can be adapted to different languages such that a variety of users can use the application.

REFERENCES

- [1] Jagtap Nilesh, Pawan Alai, Chavhan Swapnil and Bendre M.R. "Voice Based System in Desktop and Mobile Devices for Blind People". In International Journal of Emerging Technology and Advanced Engineering (IJETA), 2014 on Pages 404-407 (Volume 4, issue 2).
- [2] Ummuhanysifa U., Nizar Banu P K, "Voice Based Search Engine and Web page Reader". In International Journal of Computational Engineering Research (IJCER). Pages 1-5.
- [3] 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), 2014 on Pages 5089-5092. (Volume 3, Issue 1).
- [4] The Radicati website. [Online]. Available: <http://www.radicati.com/wp/wp-content/uploads/2014/01/Email>.

APPENDIX

A. Source code

```
import speech_recognition as sr
import pyttsx3
import datetime
import wikipedia
import webbrowser
import os
import time
import subprocess
import wolframalpha
import json
import requests
import smtplib
from email.message import EmailMessage
```

In[2]:

```
engine=pyttsx3.init('sapi5')
voices=engine.getProperty('voices')
engine.setProperty('voice',voices[1].id)
rate=engine.getProperty('rate')
engine.setProperty('rate',160)
```

In[3]:

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

In[4]:

```
def wishMe():
    hour=datetime.datetime.now().hour
    if hour>=0 and hour<12:
        speak("Hello,Good Morning")
        print("Hello,Good Morning")
    elif hour>=12 and hour<18:
        speak("Hello,Good Afternoon")
        print("Hello,Good Afternoon")
```

In[5]:

```
def takeCommand():
    r=sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        speak("listening...")
        audio=r.listen(source)

    try:
        statement=r.recognize_google(audio,language='en-in')
        print(f"user said:{statement}\n")

    except Exception as e:
        speak("Pardon me,Say again")
        return "None"
    return statement.lower()
```

In[6]:

```
def send_email(*args):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login('shivasunny451@gmail.com', '7095651014Shiva')
    email = EmailMessage()
    email['From'] = 'shivasunny451@gmail.com'
    email['To'] = args[0]
    email['Subject'] = args[1]
    if len(args)==4:
        email['Cc']=email_list[cc]
    email.set_content(args[2])
    server.send_message(email)
```

In[7]:

```
email_list =
{"shiva":"vshivasankarareddy849@gmail.com","bharadwaj":"Bharadw
ajvemuri2000@gmail.com","personal":"shivasunny451@gmail.com"}
def get_email_info():
    speak('To Whom you want to send email')
    name =takeCommand()
```

```

else:
    speak("Your Sender Mail is not in Contact....Can You type mail")
    receiver = input()
    print(receiver)
    speak('What is the subject of your email?')
    subject = takeCommand()
    print(subject)
    speak('Tell me the text in your email')
    message = takeCommand()
    print(message)
    speak('You Want to add any Cc.that is Carbon copy,then add Mail
for Carbon copy')
    ccrec=takeCommand()
    if "no" in ccrec:
        send_email(receiver,subject,message)
    else:
        cc = takeCommand()
        print(cc)
        cc=email_list[cc]
        send_email(receiver,subject,message,cc)
    speak('Hey,Your email is sent')
    speak('Do you want to send more email?')
    send_more = takeCommand()
    if 'yes' in send_more:
        get_email_info()
    else:
        speak("ThankYou,....Now we are comming out of gmail")

```

In[8]:

```

wishMe()
if __name__=='__main__':
    speak("Tell me how can I help you now?")
    while True:
        statement = takeCommand().lower()
        if statement==0:
            speak("I an not able to hear....can you say again")
            statement = takeCommand().lower()

```

```

speak('your personal assistant is shutting down,Good bye')
print('your personal assistant is shutting down,Good bye')
break

```

if "send mail" in statement or "send gmail" in statement:

```
get_email_info()
time.sleep(3)
```

if 'wikipedia' in statement:

```
speak('Searching Wikipedia...')
statement =statement.replace("wikipedia", "")
results = wikipedia.summary(statement, sentences=3)
speak("According to Wikipedia")
print(results)
speak(results)
time.sleep(3)
```

elif 'open youtube' in statement:

```
webbrowser.open_new_tab("https://www.youtube.com")
speak("youtube is open now")
speak('your personal assistant is shutting down,Good bye')
break
```

elif 'play' in statement:

```
song=command.replace('play',"")
talk('Playing')
print(song)
pywhatkit.playonyt(song)
break
```

elif 'open google' in statement:

```
webbrowser.open_new_tab("https://www.google.com")
speak("Google chrome is open now")
speak('your personal assistant is shutting down,Good bye')
break
```

elif 'open gmail' in statement:

```
webbrowser.open_new_tab("gmail.com")
speak("Google Mail open now")
time.sleep(3)
```

elif "weather" in statement:

```
api_key="8ef61edcf1c576d65d836254e11ea420"
```

```
base_url="https://api.openweathermap.org/data/2.5/weather?"
```

```

speak("whats the 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_humidiy = y["humidity"]
    z = x["weather"]
    weather_description = z[0]["description"]
    speak(" Temperature in kelvin unit is " +
          str(current_temperature) +
          "\n humidity in percentage is " +
          str(current_humidiy) +
          "\n description  " +
          str(weather_description))
    print(" Temperature in kelvin unit = " +
          str(current_temperature) +
          "\n humidity (in percentage) = " +
          str(current_humidiy) +
          "\n description = " +
          str(weather_description))

```

```

elif "open stackoverflow" in statement:
    webbrowser.open_new_tab("https://stackoverflow.com/login")
    speak("Here is stackoverflow")
    break;

```

```

elif 'news' in statement:
    news =
webbrowser.open_new_tab("https://timesofindia.indiatimes.com/home/headlines")
    speak('Here are some headlines from the Times of
India,Happy reading')
    break;

```

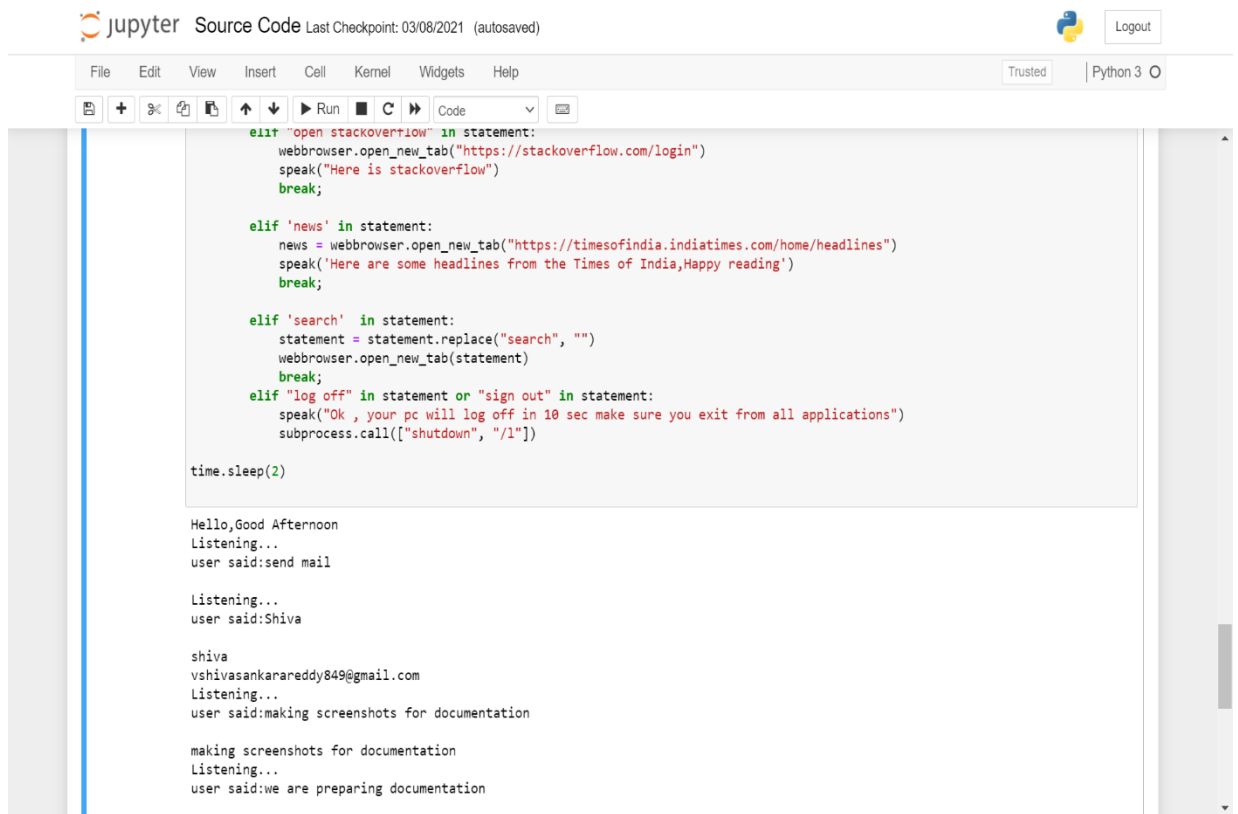
```

elif 'search' in statement:
    statement = statement.replace("search", "")
    webbrowser.open_new_tab(statement)
    break;

```

```
elif "log off" in statement or "sign out" in statement:  
    speak("Ok , your pc will log off in 10 sec make sure you exit  
from all applications")  
    subprocess.call(["shutdown", "/l"])  
  
time.sleep(2)
```

B.screen shots



The screenshot shows a Jupyter Notebook interface. At the top, there's a header with the Jupyter logo, 'Source Code', and 'Last Checkpoint: 03/08/2021 (autosaved)'. On the right, there's a 'Logout' button. Below the header is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. To the right of the menu bar are 'Trusted' and 'Python 3' indicators. The main area contains a code cell with the following Python code:

```
elif "open stackoverflow" in statement:
    webbrowser.open_new_tab("https://stackoverflow.com/login")
    speak("Here is stackoverflow")
    break;

elif 'news' in statement:
    news = webbrowser.open_new_tab("https://timesofindia.indiatimes.com/home/headlines")
    speak('Here are some headlines from the Times of India,Happy reading')
    break;

elif 'search' in statement:
    statement = statement.replace("search", "")
    webbrowser.open_new_tab(statement)
    break;

elif "log off" in statement or "sign out" in statement:
    speak("Ok , your pc will log off in 10 sec make sure you exit from all applications")
    subprocess.call(["shutdown", "/1"])

time.sleep(2)
```

Below the code cell, the output is displayed as a series of text lines:

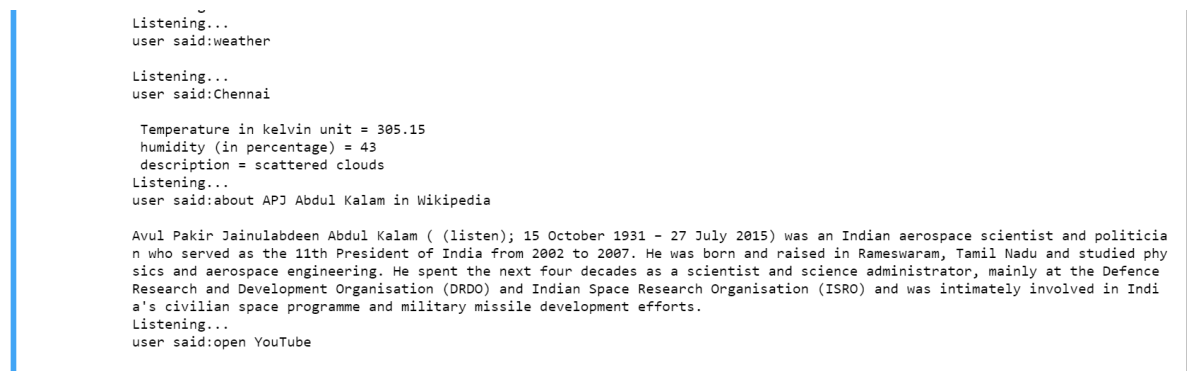
```
Hello,Good Afternoon
Listening...
user said:send mail

Listening...
user said:Shiva

shiva
vshivasankanareddy849@gmail.com
Listening...
user said:making screenshots for documentation

making screenshots for documentation
Listening...
user said:we are preparing documentation
```

Figure: Output for sending mail.



The screenshot shows a Jupyter Notebook interface. The code cell contains the following Python code:

```
~
Listening...
user said:weather

Listening...
user said:Chennai

Temperature in kelvin unit = 305.15
humidity (in percentage) = 43
description = scattered clouds
Listening...
user said:about APJ Abdul Kalam in Wikipedia

Avul Pakir Jainulabdeen Abdul Kalam ( (listen); 15 October 1931 – 27 July 2015) was an Indian aerospace scientist and politician who served as the 11th President of India from 2002 to 2007. He was born and raised in Rameswaram, Tamil Nadu and studied physics and aerospace engineering. He spent the next four decades as a scientist and science administrator, mainly at the Defence Research and Development Organisation (DRDO) and Indian Space Research Organisation (ISRO) and was intimately involved in India's civilian space programme and military missile development efforts.
Listening...
user said:open YouTube
```

Figure: Output for getting weather.

C. PLAGIARISM REPORT

VOICE BASED MAIL ASSISTANT *by* India 2020

Submission date: 16-Mar-2021 11:55PM (UTC-0700)

Submission ID: 1524801145

File name: Voice_Based_Mail_Assistant_Paper_IEEE_format.docx (90.19K)

Word count: 1557

Character count: 8445

VOICE BASED MAIL ASSISTANT

R.M Gomuthi
Department of Information
Technology
Sathyabama Institute of Science
and Technology
Chennai, India
gomssrm@gmail.com

Shiva Sankara Reddy.V
Department of Information
Technology
Sathyabama Institute of Science
and Technology
Chennai, India
vshivasankarareddy8499@gmail.com

Bharadwaj.V
Department of Information
Technology
Sathyabama Institute of Science
and Technology
Chennai, India
bharadwajvemuri2000@gmail.com

ABSTRACT:

The Internet is perhaps the main types of extravagance in regular daily existence. Everybody utilizes realities and data on the Internet. Daze individuals, then again, will think that its hard to get to message assets. The improvement of PC programs has paved up plenty chances for the outwardly users. Contingent upon the visual climate that reacts to sound, screen perusers can assist many visually impaired individuals with getting to Internet programs. This venture presents a voice message program, which can be utilized by daze individuals to effortlessly get to email. Having part in research encourages daze people to receive and send the messages related to voice-based instantly. This can be done with the help of computer assistance in their own or home language.

Objectives:

- The primary motivation behind our application is to assist dazzle individuals with sending and get mail without the assistance of any other individual.
- Users need to talk pre-characterized watchwords to play out each undertaking, for example, login, send, cc, bcc, erase, and so forth
- Along these lines, it will be a lot simpler for undetectable individuals to discover the mail. In this way, the need to depend on others to utilize email will be diminished

- The total program depends on SIT (Speech-To-Text), TTS (Text-To - Speech), and IVR (Interactive Voice Response).

INTRODUCTION:

Web correspondence innovation furnishes us with freedoms to speak with the world through email, video conferencing, texting, and so forth However, screens can't utilize these zones, as they all require visual keenness.

This task presents an application that is broadly utilized by dazzle individuals to get phone message, and numerous other web applications that run by voice orders.

The run of the mill composing velocity of common individuals is 40 words each moment, while the average composing rate of visually impaired individuals is just 32 words.

LITERATURE SURVEY:

Author: Ms. Archana A. Nikose, March 2019

We offer a voice message advancement program that visually impaired individuals can use to effectively get to mixed media administrations, like email.

Pros: System. However long the Web framework is utilized appropriately, it very well may be supposed to be completely open.

Cons: All capacities are upheld and can be

handily made with a solitary mouse snap, and clients of any style can without much of a stretch use it.

Author: Ruchi Khedekar Oct -2019

In this article, we are proposing a program that will help the outwardly hindered to get to email benefits all the more successfully.

Pros: Client show layout is constructed utilizing Adobe Dreamweaver CS3.

Cons: Likewise utilize the mouse to snap to track down the relating administration.

Author: Prof. Ila Sawant April 2019

This article intends to improve the email framework that can assist dazzle individuals with utilizing the application to convey in any event, when there is no noticeable inability.

Pros: Framework totally depends on the given voice input which is intuitive, This helps the user to easily understand the framework and it is proficient.

Cons: Use mouse usefulness rather than console to change voice over to message.

EXISTING SYTEM:

By 2014, an aggregate of 4.1 billion email accounts were made, and before the finish of 2018, it is assessed that there will be 5.2 billion records. [4] This makes email the most broadly utilized strategy for correspondence. Individuals confronting visual difficulties can't utilize the most well-known postal administrations we use in our everyday lives. This is said on the grounds that they did not give any area to key work force to tune in to the screen content. Since they were not able to perceive what's now on the screen, they will not be able to sort out where to snap the play out of the ideal undertaking. [3] For individuals who are confronting the test of utilizing a PC interestingly, albeit this is not difficult to utilize, it's anything but a simple errand for customary clients. Despite the fact that there are many screen perusers, the

troubles these individuals face are negligible. Screen perusers read anything on the screen, and in light of the fact that screen perusers can't follow mouse space, individuals need to utilize console alternate ways to play out these activities. This implies two things; if the cursor position isn't followed, a client can utilize the mouse pointer since it won't be hindered by any stretch of the imagination, and besides, the client ought to be acquainted with the console where each key is found. Consequently, another client on the PC can't utilize this help since it doesn't have a clue about the key area. Another backlink utilized is that screen perusers read the substance in a successive way, so clients can just make screen content in fundamental HTML design as it were. Subsequently, a recently evolved website page that doesn't follow this worldview will make the webpage simpler to utilize, and will just motivation more issues for certain individuals. All these are a portion of the issues that the current framework will defeat in the framework we are creating

PROPOSED SYSTEM:

The arranged framework depends on another framework and isn't pretty much as clear as the available mail framework. In improving admittance to the arranging framework, an indestructible component of the past important in the mind. The current program can't furnish you with this comfort. Along these lines, the neighborhood advancement pattern of the current framework is totally conflicting to the framework. As opposed to the current framework

The arrangement puts more accentuation on client straightforwardness for unpracticed clients, and the arrangement centers around youngsters with different kinds of clients, including the outwardly weakened and the outwardly debilitated and unskilled. The entire construction depends on IVR's intelligent voice reaction. In the event that

you utilize this program, the PC will advise the customer to play out a particular errand to get the suitable help, and if the customer needs to get to the fitting assistance, they need to play out the capacity. Perhaps the main advantages of this program is t the client doesn't have to use a console. All exercises will be available on voice measures.

Advantages:

- We can use the mail service with ease.
- Disabled people could operate hassle free.
- It is accurate and highly efficiency.

METHODOLOGY:

- We have implemented a interactive chat bot which runs through voice commands.

After developing chatbot we have created module for sending mail through smtplib library here we can access many features like Cc,Subject,Text entirely through voice commands.

Here chatbot has many commands to run web applications such as YouTube, google, weather in current city, search information in google engine, search about a person have a brief description about 3-5 lines spelled through voice.

We have made a user interface where voice commands are displayed in message box.

ARCHITECTURE DIAGRAM:

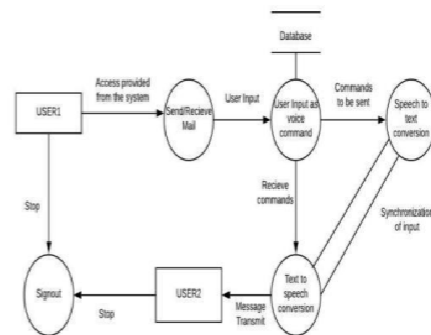


FIG:1 ARCHITECTURE DIAGRAM

PROJECT IMPLEMENTATION:

- **SMTPLIB:** - A basic mail move convention. The SMTPLIB module characterizes a SMTP customer meeting thing, which can be utilized to send email to any Internet gadget with a SMTP audience.
- **Py Audio:** - The binding for port audio is provided by Py audio and cross-platform audio input and output library. We can record and play audio on different platforms using python.
- **Pytsx3:** - pyttsx3 is a library for changing over text into a Python. In contrast to different libraries, it can work disconnected and is viable with Python 2 and 3.
- **Speech acknowledgment:** - Lords for discourse acknowledgment, uphold for various web crawlers and APIs both on the web and disconnected.
- **Python**
 - I. Mediator Modules in action.
 - II. Extensive uphold Libraries.
 - III. Open Source and Community Development.
 - IV. Productivity with Speed.

Flow chart:

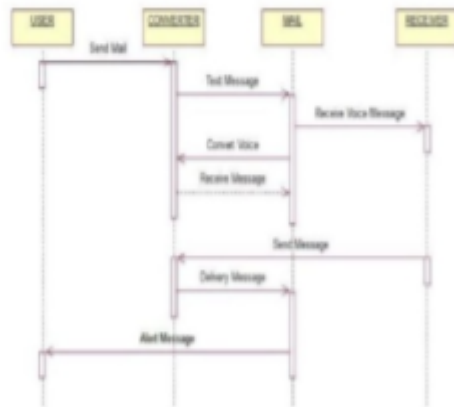


FIG:2 SEQUENCE DIAGRAM

FUTURE WORK AND CONCLUSION:

- This mail system based on voice is very effective and helpful for people who are visually impaired. This system also has the ability to convert speech to text and text to speech with the help of speech reader which is useful for blind people as well as visually impaired person. .
- The application will detect the voice commands of user and with the help of interactive voice response user can interact with the application easily and efficiently. The user only need to speak the predefined keywords for each action to take place such as Login, Send, CC, BCC, Delete, etc.
- To make the application more secure encryption and decryption algorithms can be used to increase the users trust while logging in. The algorithm can be made smart .

Output and screenshots:



```

hello,good Evening
Listening...
user said:send mail

Listening...
user said:Shiva

shiva
vshivasankaranarajy@44@gmail.com
Listening...
user said:making demo video

making demo video
Listening...
user said:we have completed 50% of work

we have completed 50% of work
    
```

FIG:3 OUTPUT

REFERENCES:

- [1] Jagtap Nilesh, Pawan Alai, Chavhan Swapnil and Bendre M.R. "Voice Based System in Desktop and Mobile Devices for Blind People". In International Journal of Emerging Technology and Advanced Engineering (IJETA), 2014 on Pages 404-407 (Volume 4, issue 2).
- [2] Ummuhanyisa U.Nizar Banu P K , "Voice Based Search Engine and Web page Reader". In International Journal of Computational Engineering Research (IJCER). Pages 1-5.
- [3] 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), 2014 on Pages 5089-5092. (Volume 3, Issue 1).
- [4] The Radicati website. [Online]. Available: <http://www.radicati.com/wp/wp-content/uploads/2014/01/Email>.

VOICE BASED MAIL ASSISTANT

ORIGINALITY REPORT

7 %

6%

INTERNET SOURCES

2%

PUBLICATIONS

2%

STUDENT PAPERS

SIMILARITY INDEX

PRIMARY SOURCES

1

docplayer.net

Internet Source

2

content.iospress.com

Internet Source

3

www.journals.resaim.com

Internet Source

4

www.ij sar.in

Internet Source

5

core.ac.uk

Internet Source

6

Sherly Noel. "Human computer

interaction(HCI) based Smart Voice

Email (Vmail) Application -

Assistant for Visually Impaired

Users (VIU)", 2020 Third

International Conference on Smart

Systems and Inventive Technology

(ICSSIT), 2020

Publication

2%

2%

1%

1%

1%

<1%

