

# **INTERACTIVE VOICE BASED EMAIL FOR BLIND PEOPLE**

## **A PROJECT REPORT**

*Submitted by*

<b>KARAN. M</b>	<b>(820519104016)</b>
<b>PRASANTH. S</b>	<b>(820519104023)</b>
<b>RAJA . S</b>	<b>(820519104025)</b>
<b>SANTHOSH. S</b>	<b>(820519104028)</b>
<b>VIGNESH. S</b>	<b>(820519104036)</b>

*In partial fulfillment for the award*

*of the degree of*

**BACHELOR OF ENGINEERING**

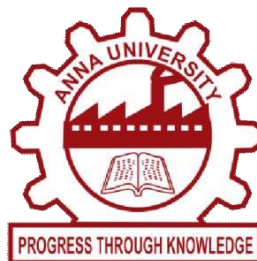
**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**ANNAI COLLEGE OF ENGINEERING AND**

**TECHNOLOGY KOVILACHERY,**

**KUMBAKONAM**



**ANNAUNIVERSITY: CHENNAI 600025**

**MAY-2023**

# **ANNAUNIVERSITY: CHENNAI 600025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**INTERACTIVE VOICE BASED EMAIL FOR BLIND PEOPLE**” is the bonafide work **KARAN . M (820519104016)** **PRASANTH S (820519104023)**, **RAJA S (820519104025)**, **SANTHOSH.S (820519104028)** ,**VIGNESH S (820519104036)** who carried out the project work under my supervision.

### **SIGNATURE**

**Mrs.A.N.TAMILARASI, M.TECH.,**  
**HEAD OF THE DEPARTMENT**  
Department of CSE  
Annai College of Engg &Tech,  
Kovilacheri-612503

### **SIGNATURE**

**Mrs.K.GUNA, M.E.,**  
**ASSISTANT PROFESSOR**  
Department of CSE  
Annai College of Engg &Tech,  
Kovilacheri-612503

Submitted for Anna University viva-voce examination held on  
.....

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## DECLARATION

We declare that the work entitled “**INTERACTIVE VOICE BASED EMAIL FOR BLIND PEOPLE**” is submitted in partial fulfillment of the requirement for the award of degree in BE/CSE., ANNA UNIVERSITY OF TECHNOLOGY, CHENNAI, is a record of my own work carried out by me during the academic year 2019-2023 under the supervision and guidance of **Mrs.K.GUNA M.E.**, Assistant professor of computer science & Engineering, Annai college of engineering & technology. The extent and source of information are derived from existing literature have been indicated through the dissertation the appropriate places. The matter embodied in this work is original and has been submitted for the award of any degree or diploma, either in this or any university.

Name	Register Number	Signature
KARAN . M	820519104016	
PRASANTH . S	820519104023	
RAJA . S	820519104025	
SANTHOSH . S	820519104028	
VIGNESH . S	820519104036	

I certify that the declaration made above by the candidate is true.

Signature of Guide,

**Mrs.K.GUNA,M.E.,**  
**ASSISTANT PROFESSOR**  
Department of CSE

## ACKNOWLEDGEMENT

Everything is possible by the way of creator. So the first of all thank the almighty for giving knowledge

We thank our honorable chairman **Mr.M.I.HUMAYNKABIR**, Annai College of Engineering and Technology, managed for having given us the opportunity to do the course..

We are much pleased to acknowledgment our indebtedness to our beloved principal **Dr.J.PRAKASH ARUL JOSE. Ph.D.**, for his great appreciation and endorsement.

Our sincere thanks to our Head of the Department of Computer science & Engineering. **Mrs.A.N.TAMILARASI,M.TECH.**, or the enormous facilities provided to complete our project work in time and also thanks

forhervalueableguidance,supportandencouragementduringtheprojectreviews. Our profound gratitude to our graceful project coordinator

**Mrs.K.GUNA,M.E.**,whoseencouragementandsupportenabledthisprojectmaterializeand contributed to its success.

We wish to express our thanks and profound gratitude to our project guide **Mrs.K.GUNA, M.E.**, Annai College of Engineering and constant Encouragement, which has stimulated us to make are mark able project

## **ABSTRACT**

Internet has become one of the basic amenities for day-to-day living. Every human being is widely accessing the knowledge and information through internet. However, blind people face difficulties in accessing these text materials, also in using any service provided through internet. The advancement in computer based accessible systems has opened up many avenues for the visually impaired across the globe in a wide way. Audio feedback based virtual environment like, the screen readers have helped Blind people to access internet applications immensely. This project describes the Voicemail system architecture that can be used by a Blind person to access e-Mails easily and efficiently. The contribution made by this project has enabled the Blind people to send and receive voice-based e-Mail messages in their native language with the help of a computers. The objective of Voice Based Email for Visually Impaired is to help challenge one's access mails easily and efficiently. This application is based on using speech-to-text and text-to-speech converters, thus enabling everyone to control their mail accounts using their voice only and be able to read, send, and perform all the other useful tasks. The system will prompt the user with voice commands to perform certain action and the user will respond to the same.

## CONTENTS

<b>Chapter No.</b>	<b>Title</b>	<b>Page No.</b>
	Acknowledgement	iii
	Abstract	iv
	List of Figures	ix
	List of Tables	x
<b>1</b>	<b>INTRODUCTION</b>	<b>10</b>
<b>1.1</b>	Introduction	10
<b>2</b>	<b>LITERATURESURVEY</b>	<b>12</b>
<b>3</b>	<b>SYSTEMSTUDY</b>	<b>19</b>
<b>3.1</b>	Existing System	19
<b>3.2</b>	Proposed System	19
<b>3.3</b>	Architecture	21
<b>3.4</b>	Data Flow Diagram	22
<b>3.5</b>	Data Flow Diagram	23
<b>3.6</b>	ER Diagram	25
<b>3.7</b>	UML Diagram	26
<b>3.8</b>	Use Case Diagram	26
<b>4</b>	<b>MODULE DISCRIPTION</b>	<b>31</b>
<b>5</b>	<b>SYSTEM SPECIFICATION</b>	<b>51</b>
<b>5.1</b>	Hardware Requirements	51
<b>5.2</b>	Software Requirements	51
<b>5.3</b>	Software Description	52
<b>5.4</b>	Hardware Description	58

5.5	Testing Procedure	60
<b>6</b>	<b>CONCLUSION</b>	<b>63</b>
<b>7</b>	<b>REFERENCE</b>	<b>64</b>

## **LIST OF FIGURES**

<b>FIGURE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
1	Existing System	21
2	CONCEPTDIAGRAM	23
3	ARCHITECTUREDIAGRAM	24
4	DATAFLOWDIAGRAM	26
5	ER Diagram	27
6	UML Diagram	31
7	Class Diagram	32
8	Activity Diagram	33
9	Sequence Diagram	34
10	Collaboration Diagram	35



## **LIST OF TABLES**

<b>Table No.</b>	<b>Title</b>	<b>Page No.</b>
11	Data flow Symbols	25
12	ER Diagram Symbol	27

## **LIST OF SYMBOLS**

STT	Speech To Text
TTS	Text To Speech
SMTP	Simple Mail Transfer Protocol
UML	Unified Modeling Language
GUI	Graphical User Interface
GPL	General Public License
NLP	Natural Language Processing

# 1.INTRODUCTION

## 1.1Introduction:

In today's world, the combination of technology and the Internet has made communication so easy. In 2015, the world's email user base was almost 2.6 billion. By 2025, the world's email users globally will increase to over 4 billion, using e-mail as the most frequently used type of communication. There are an estimated 285 million people with sight loss around the world with 40 million blind and 245 million visually impaired. However, using this technology is quite hard for people with visual deficiencies. For sending an email they need to tell the full content of the letter to any other person (who is not sight-impaired), and then the third person will compose the letter and send it on a blind man's behalf. However, that is not the correct solution. Existing technologies such as screen readers, automatic speech recognition, text to speech, and speech to text made things easier for the visually impaired, although partially. As a result, the need arose to create a complete voice-based application that could send or receive emails or perform messaging related operations.

The application which is able to be a python-based application provide a voice-based mailing service where they might read and send mail on their own through their g-mail accounts with none other person's help or guidance. The applying may be utilized by near-blind to access mails easily. The applying will use text to speech and speech to text as alternative of keyboard. The application involves using Interactive Voice Response (IVR) which provides the user to manage their mail accounts through their voice only. Through voice, they are going to be ready to read, send and perform all the opposite required tasks. The system traces speech at run time through a microphone and processes the speech to acknowledge the relevant text. The most advantage of this method is that it eliminates the utilization of keyboard and therefore the user can access the system though voice only.

Internet has become one of the basic amenities for day-to-day living. Every human being is widely accessing the knowledge and information through internet. However, blind people face difficulties in accessing these text materials, also in using any service provided through internet. The advancement in computer based accessible systems has opened up many avenues for the visually impaired across the globe in a wide way. Audio feedback based virtual environment like, the screen readers have helped Blind people to access internet applications immensely. We describe the Voicemail system architecture that can be used by a Blind person to access e-Mails easily and efficiently. The contribution made by this research has enabled the Blind people to send and receive voice based e-Mail messages in their native

language with the help of a computer

We have seen that the introduction of Internet has revolutionized many fields. Internet has made life of people so easy that people today have access to any information they want easily. Communication is one of the main fields highly changed by Internet. E-mails are the most dependable way of communication over Internet, for sending and receiving some important information. But there is a certain norm for humans to access the Internet and the norm is you must be able to see. But there are also differently abled people in our society who are not gifted with what you have. There are some visually impaired people or blind people who can't see things and thus can't see the computer screen or keyboard. A survey has shown that there are more than 240 million visually impaired people around the globe. That is, around 240 million people are unaware of how to use Internet or E-mail. This system aims at developing an email system that will help even a visually impaired person to use the services for communication without previous training. The system is completely built on interactive voice response which will make it user friendly and efficient to use. The entire project is based on voice interaction which means speech recognition and synthesis.

Internet plays a vital role in today's world of communication. Today the world is running on the basis of internet. No work can be done without use of internet. Electronic mail i.e. email is the most important part in day to day life. But some of the people in today's world don't know how to make use of internet, some are blind or some are illiterate. So it goes very difficult to them when to live in this world of internet. Nowadays there are various technologies available in this world like screen readers, ASR, TTS, STT, etc. but these are not that much efficient for them. Around 39 million people are blind and 246 people have low vision and also 82 of people living with blindness are 50 aged and above. We have to make some internet facilities to them so they can use internet. Therefore we came up with our project as voice based email system for blinds which will help a lot to visually impaired peoples and also illiterate peoples for sending their mails. The users of this system don't need to remember any basic information about keyboard shortcuts as well as location of the keys. Simple mouse click operations are needed for functions making system easy to use for user of any age group. Our system provides location of where user is prompting through voice so that user doesn't have to worry about remembering which mouse click operation he/she wants to achieve.

## **2.LITERATURE SURVEY**

### **1. Voice Based Email for Blind**

**Year:**2022

**Author:** Malavika Arun1, Meera M Nair, Nivya Raj, Svetlana Bright

#### **Abstract:**

The internet has now become one of the most important tools for everyday life. Every human being uses the internet to gain access to knowledge and information, as well as to communicate. During this modern technological era, there are numerous ways to communicate with others over the internet. Many of them are opting for the most effective mode of communication, which is email correspondence (Email). However, blind persons have difficulty accessing these internet-based resources, as well as using any internet-based service. Audio-based virtual environments, including screen readers and a slew of voice-based search engines, have made it much easier for blind persons to use the internet. This study tries to explain the architecture of a Voice-based mail system that a blind person can use to access emails rapidly and effectively. This research's contribution has enabled blind persons to send and receive voice-based e-Mail messages with the aid of a computer.

#### **Introduction:**

People are becoming increasingly used to digital life and communication as technology advances. In this highly evolved era, there are several ways to contact others over the internet. The internet has become the bedrock of modern society. The most important component of daily life is email, or electronic mail. Email is a piece of software that allows users to communicate with others by sending emails and aids in corporate communication.

#### **Advantages:**

- Blinds are especially useful if you have windows that are facing an open street or right across from your neighbour's windows.
- Adding blinds may help you feel more secure, allowing you to keep away from

prying eyes.

**Disadvantages:**

- On the other hand, blinds will need to be cleaned more often. In addition, blinds are less efficient at keeping in the heat

## **2. VOICE-BASED EMAIL SYSTEM FOR VISUALLY IMPAIRED PEOPLE**

**Year:**2022

**Author:** Rajini S ,Dhivyaa A S , Jananipriya R M , Karthikapriyaa V S

**Abstract:**

Nowadays, Technology usage has increased, and it offers many opportunities for present-day generations to fully embrace the Internet. Email continues to be the most common form of communication technology in the business world. People with vision loss find it very difficult to use this technology because their use requires visual and touch perception. Developing computerized practice systems has opened up numerous possibilities for the visually impaired. This system is very helpful for the blind to use Internet applications. This project introduces the structural design of a voicemail system that blind people can use for easy email access. The application uses speech-text and text-speech algorithms where visually impaired users can make effective use of the application.

**Introduction:**

In today's world, the combination of technology and the Internet has made communication so easy. In 2015, the world's email user base was almost 2.6 billion. By 2025, the world's email users globally will increase to over 4 billion, using e-mail as the most frequently used type of communication. There are an estimated 285 million people with sight loss around the world with 40 million blind and 245 million visually impaired. However, using this technology is quite hard for people with visual deficiencies. For sending an email they need to tell the full content of the letter to any other person (who is not sight-impaired), and then the third person will compose the letter and send it on a blind man's behalf. However, that is not the correct solution. Existing technologies such as screen readers, automatic speech recognition, text to speech, and speech to text made things easier for the visually impaired, although partially. As a result, the need arose to create a complete

voice-based application that could send or receive emails or perform messaging related operations.

**Advantages:**

- For those without eyes, this voice-based email system is helpful since it enables them to understand their location

**Disadvantages:**

- They cannot see the screen of the computer and cannot use the keyboard.

## **VOICE BASED E-MAIL SYSTEM FOR THE BLIND**

**Year:**2020

**Author:**Vedant Chidgopkar, Suraj Jadhav, Atharva Joshi, Abhishek Khedekar

**Abstract:**

The visually challenged people find it very difficult to access the technology because of the fact that using them requires visual perception. Unlike normal users they require some practice for using the available technologies. This application aims at developing an email system that will help even a visually impaired person to use the services for communication without prior training. . It has been observed that nearly about 60% of total blind population across the world is present in INDIA. In this paper, we describe the voice mail architecture used by blind people to access E-mail. This architecture will also reduce cognitive load taken by blind to remember and type characters using keyboard. It also helps handicapped and illiterate people. The system will not let the user make use of keyboard instead will work only on mouse operation and speech conversion to text. Also this system can be used by any normal person also for example the one who is not able to read. The system is completely based on interactive voice response which will make it user friendly and efficient to use

## **Introduction:**

We have seen that the introduction of Internet has revolutionized many fields. Internet has made life of people so easy that people today have access to any information they want easily. Communication is one of the main fields highly changed by Internet. E-mails are the most dependable way of communication over Internet, for sending and receiving some important information. But there is a certain norm for humans to access the Internet and the norm is you must be able to see. But there are also differently abled people in our society who are not gifted with what you have. There are some visually impaired people or blind people who can't see things and thus can't see the computer screen or keyboard. A survey has shown that there are more than 240 million visually impaired people around the globe. That is, around 240 million people are unaware of how to use Internet or E-mail. The only way by which a visually challenged person can send an E-mail is, they have to speak the entire content of the mail to another person ( not visually challenged ) and then that third person will compose the mail and send on the behalf of the visually challenged person. But this is not a right way to deal with the problem. It is very unlikely that every time a visually impaired person can find someone for help. The aim of this project is to make differently able use the system efficiently and pave a way for differently able to easily access their emails .This system aims at developing an email system that will help even a visually impaired person to use the services for communication without previous training. The system is completely built on interactive voice response which will make it user-friendly and efficient to use. The entire project is based on voice interaction which means speech recognition and synthesis. Voice technology is being used in recent times proving to help users to have easy access to their respective applications or websites. We understand how vulnerable the visually impaired are in today's digital world. Hence, by this project, we will strive to develop a useful project by using speech recognition and synthesis and thus providing a better solution to their crisis. The study has found that generally there are limited resources for the disabled. Hence, our project will be efficient and productive for the differently able automated method of recognizing an individual by means of comparing the feature vector derived from the behavioral and the physiological distinctiveness such as fingerprint, iris, face recognition etc. Face detection and recognition is becoming increasingly important. There are multiple cues available and can be used as features. Facial features are extracted and compared using support vector machine algorithm.

**Advantages:**

- On the other hand, people need to remember mouse clicks and keyboard shortcuts

**Disadvantages:**

- It can be very difficult for voice assistants to comprehend words that are said that are not in the dictionary

**Voice Based E-mail System for Visually Impaired: A Review**

**Year:**2020

**Author:** Parkhi Bhardwaj, Gunjan Sethi

**Abstract:**

As the technology is enhancing, people are coming more closer to digital life and digital communication. There are many ways to communicate with others through internet in this new advanced era. Most of them are choosing the easiest way of communication i.e., Electronic mail (E-mail). E-mail is the technology that enables user to contact with others by sending mails and also helps in business world communication. There are people who cannot use these technologies because either they are illiterate or do not have ability to see the screen. So, to make this technology closer to visually challenged people, authors proposed a Voice Based E-mail System. This system provides them the facility of communication and make them much stronger and independent. This architecture will help blind people to access e-mail and other multimedia functions. Leaving behind the old techniques, this voice-based email system will be containing new technologies that will be easily acceptable by visually challenged people

**Introduction:**

Internet is the most essential part in today's world of communication. Emails are further important way of communication that widely used in the business world. This technology has been useless for the incapacitated and oblivious people. There are around 260 million visually challenged people around the globe according to a survey. That also means these people are unaware of how to use internet or e-mail and about new advanced technologies. The solution to this problem has been in a way that either there should be third person which is not visually challenged to help them in reading and sending mails or to use screen-readers, braille keywords etc. Braille keyboards proved to be useful but not sufficient enough to perform high level operations. These ways are not correct to deal with this problem and also had drawbacks. This application uses text to speech (TTS) and speech to text (STT) converters so that visually challenged people can operate the system easily. This system reduces the complexity to remember the characters or information regarding keyboard shortcuts. Every function will be based on simple voice commands so that those people could easily make use of the technology. The model will also help blind people to become independent and strong as they will be able to communicate without the help of third person.



**Advantages:**

- For those without eyes, this voice-based email system is helpful since it enables them to understand their location.
- On the other hand, people need to remember mouse clicks and keyboard shortcuts. With this method, the usability of all users, including those who are typically blind, is given more priority

**Disadvantages:**

- It can be very difficult for voice assistants to comprehend words that are said that are not in the dictionary

**VOICE BASED EMAIL SYSTEM FOR BLIND AND HANDICAPPED PEOPLE**

**Year:**2022

**Author:** KhizarBagban, Aniket Bali, Akash Shendge , Savita Adhav

**Abstract:**

web internet is one amongst the fundamental luxury for daily living. every body is exploitation the facts and knowledge on web. On the opposite hand, blind person face problem in accessing the text resources. the event in pc based mostly handy systems has opened various opportunities for the visually disabled across. Audio response based mostly virtual surroundings, the screen readers square measure helps blind person plenty to use web applications. This project introduces the Voicemail system structural style that may be employed by a blind man to access E-Mails simply. The involvement of analysis helps blind individual to send and receive voice based mostly mails messages in their someone language with the assistance of a laptop or pc

**Introduction:**

We have seen that the beginning of web has dramatically revolutionized several fields. web has created lifetime of people really easy that individuals nowadays have access to any data they need sitting at their home. one amongst the most fields that web has revolutionized is communication. And talking concerning communication over web, the primary issue that comes in our mind is E-mail. E-mails square measure thought of to be the foremost reliable manner of communication over web, for causing or receiving some vital data. however there's a special criteria for humans to access the net and also the criteria is you want to be ready to see. you want to be thinking that what variety of criteria is that this, all with eyes will see. however there are specially abled person in our society United Nations agency aren't talented with what you have got. affirmative there square measure some visually impaired person or blind people that can't see things and so can't see the pc screen or keyboard. A survey shows that there square measure over 250 million visually impaired person round the globe. That is, around 250 million peoples measurely unaware of the way to use web or E-mail. the sole manner by that a visually impaired person will send associate E-mail is, they need to dictate the complete content of the mail to a 3rd person not visually impaired ) then the person can compose the mail and ship the behalf of the visually impaired person..

**Advantages:**

- They cannot see the screen of the computer and cannot use the keyboard. So they cannot access the internet and becomes very difficult to communicate by using the internet.

**Disadvantages:**

- On the other hand, blinds will need to be cleaned more often. In addition, blinds are less efficient at keeping in the heat.
- The window blind slats can easily get damaged, especially if you choose aluminium or plastic blinds. Blind cords can be a hazard around young children.

### 3.SYSTEM STUDY

#### 3.1 EXISTING SYSTEM:

Simple e-mail systems are available in which only voice recognition & text-to-speech systems are accessible by remembering the keyboard shortcuts to access. The existing voice-based e-mail system has made use of IVR, Speech to text converter, Mouse click event, and Screen reader. There will be a small icon of the mic on who's clicking the user had to speak and his/her speech will be converted to text format, which the blind people would see and read also There are a total number of 4.1 billion email accounts created until 2014 and there will be estimated 5.2 billion accounts by end of 2018. 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 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

#### Disadvantages:

- Users must use a mouse connected to the computer and should perform mouse click events to send and receive emails.
- In the existing system, they have chosen Web UI as the interface for a system which is not easy for the impaired people to use.

#### 3.2 PROPOSED SYSTEM:

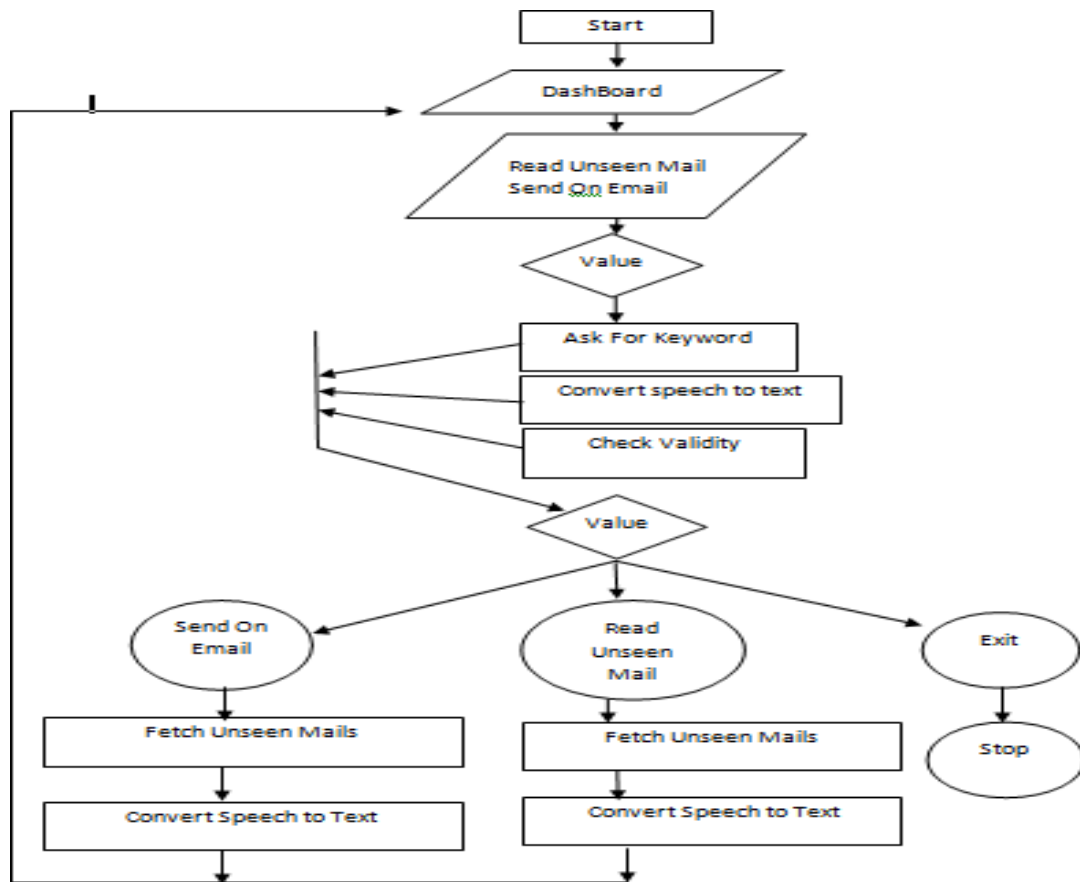
- In this proposed system it mainly concentrates on **four different types of technologies**
- Speech-to-Text (STT) this module collects the speech given by the user and converts to Text,
- Text-to-Speech (TTS) this converts the response give to system to Speech,
- Chatbot for making the conversation more sense and for giving responses more like a human and finally,

- Mail communication module for sending and receiving emails.
- The project titled voice-based email system is a web based application developed that allows blind people to use Email system easily.
- The proposed system focuses on providing the basic functionalities like composing, reading, sending and receiving emails along with voice-based interaction.
- This facilitates working with each of the above features as well as the provision for sending text as well as voice based email.
- The proposed system allows blind people to use Email system easily. As the input to the system does not use keyboard or mouse, users can easily give input by speaking the message.
- Thus, the system that we are developing is entirely different from the existing ones. Unlike other systems which focus only on a particular set of people, our system is focused on visually challenged people too

#### **Advantages:**

- Doesn't need any mouse click events to send and receive emails.
- Purely based on the voice commands given by the user.
- Chatbot is used to make the conversation smooth and more like a human response
- User friendly (as Blind person can easily use web based application). Easy Storage of data.
- More efficient.
- Requires less effort and time
- The system that we are developing is entirely different from the existing ones. Unlike other systems which focus only on a particular set of people, our system is focused on visually challenged people too. It also helps handicapped and illiterate people.





### 3.3 ARCHITECTURE DIAGRAM:



### 3.5 DATA FLOW DIAGRAM

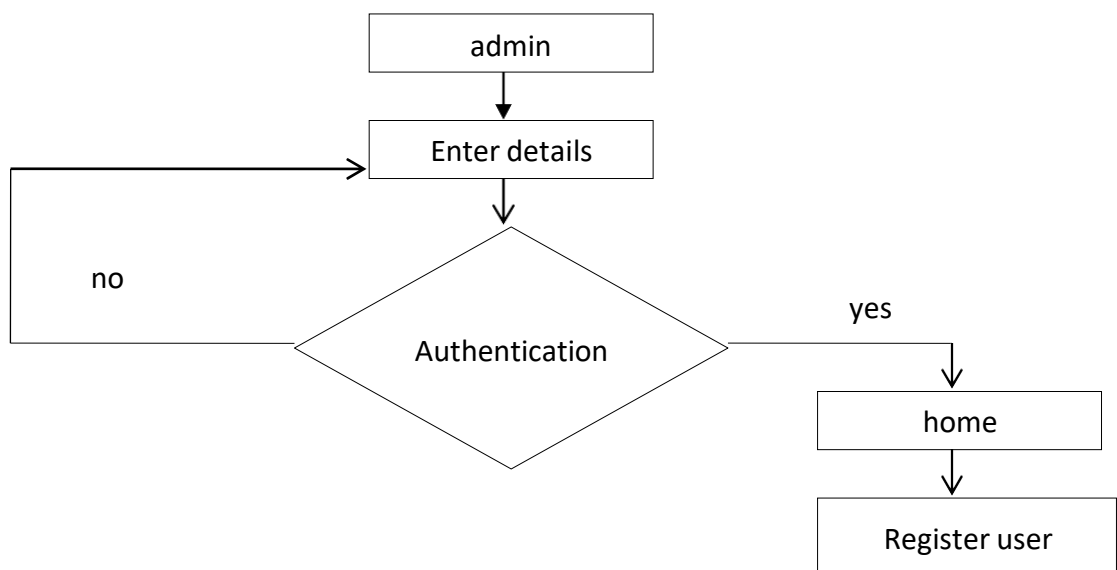
A two-dimensional diagram that explain show data is processed and transfer red in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a dataflow diagram must (1) identify external inputs and outputs, (2) determine how the input sand outputs relate to each other, and (3) explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

**Table:3.1DataflowSymbols**

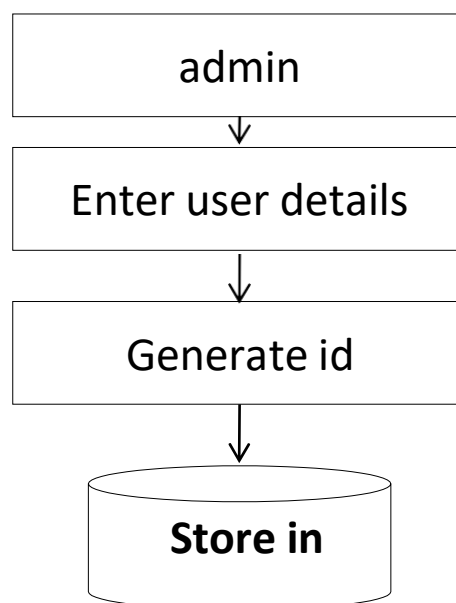
Symbol	Description
	An <b>entity</b> . A source of data or a destination for data.
	A <b>process</b> or task that is performed by the system.
	A <b>data store</b> , a place where data is held between processes.
	A <b>data flow</b> .

### 3.5 DATA FLOW DIAGRAM:

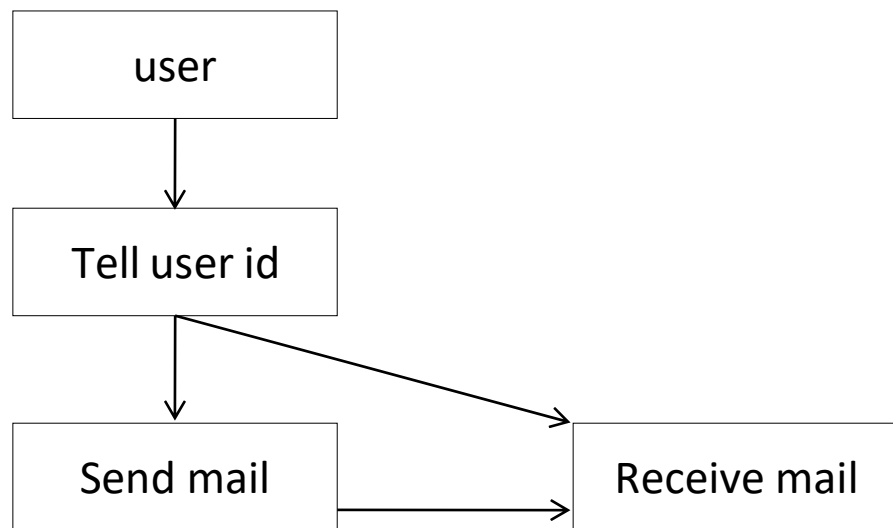
#### Level0



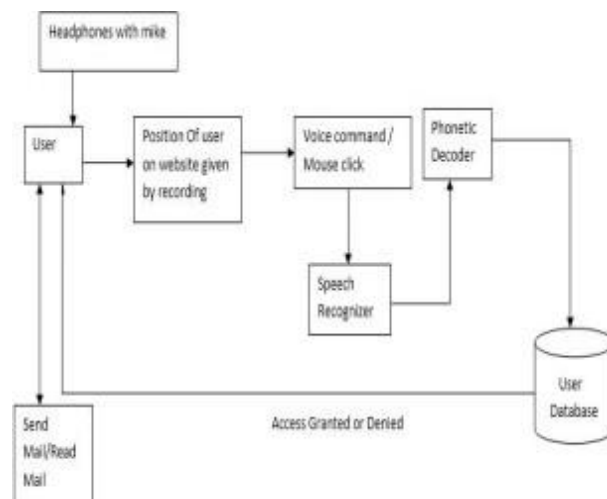
#### Level1



## Level2



## Level3

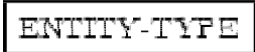

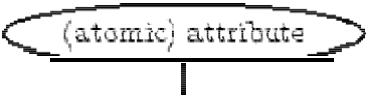
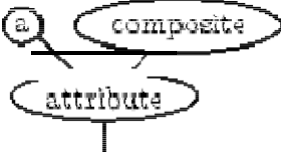
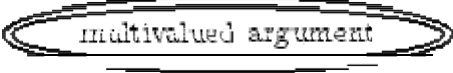





### 3.6 ER DIAGRAM

An entity-relationship model is the result of using a systematic process to describe and define a subject area of business data. It does not define business process; only visualize business data. The data is represented as components (entities) that are linked with each other by relationships that express the dependencies and requirements between them, such as: one building may be divided into zero or more apartments, but one apartment can only be located in one building. The three schema approach to software engineering uses three levels of ER models that may be developed.

**Table 3.2. ER Diagram Symbol**

S.NO.	Diagram	Types
1.		Entity types
2.		Relationship Types
3.		Atomic attribute types
4.		Composite attribute types
5.		Multi valued attribute types
6.		Derived Attribute types

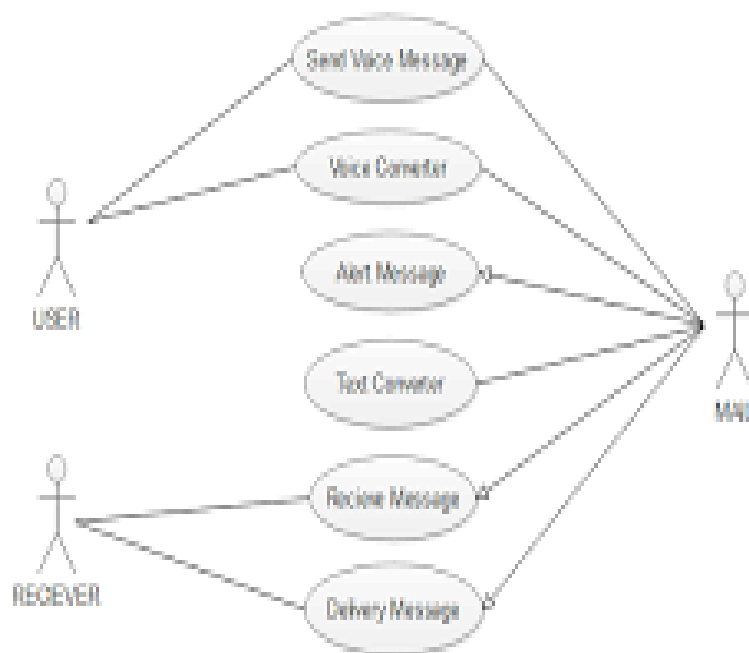
### 3.7 UML Diagram

The Unified Modeling Language (UML) is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

### 3.8 Use Case Diagram

Use case diagrams are usually referred to as behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

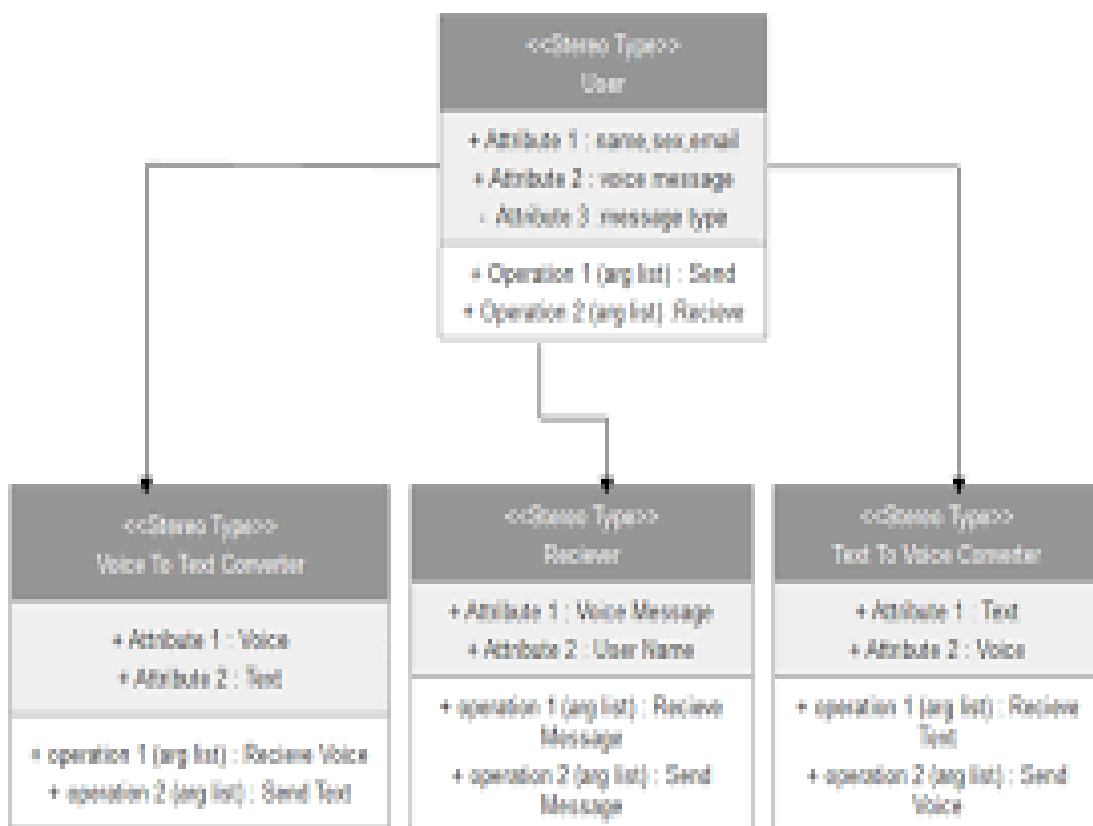
A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.



### 3.9 Class Diagram

The class diagram is the main building block of object-oriented modelling. It is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modelling.

In the diagram, classes are represented with boxes that contain three compartments:



The top compartment contains the name of the class. It is printed in bold and centered and the first letter is capitalized.

The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.

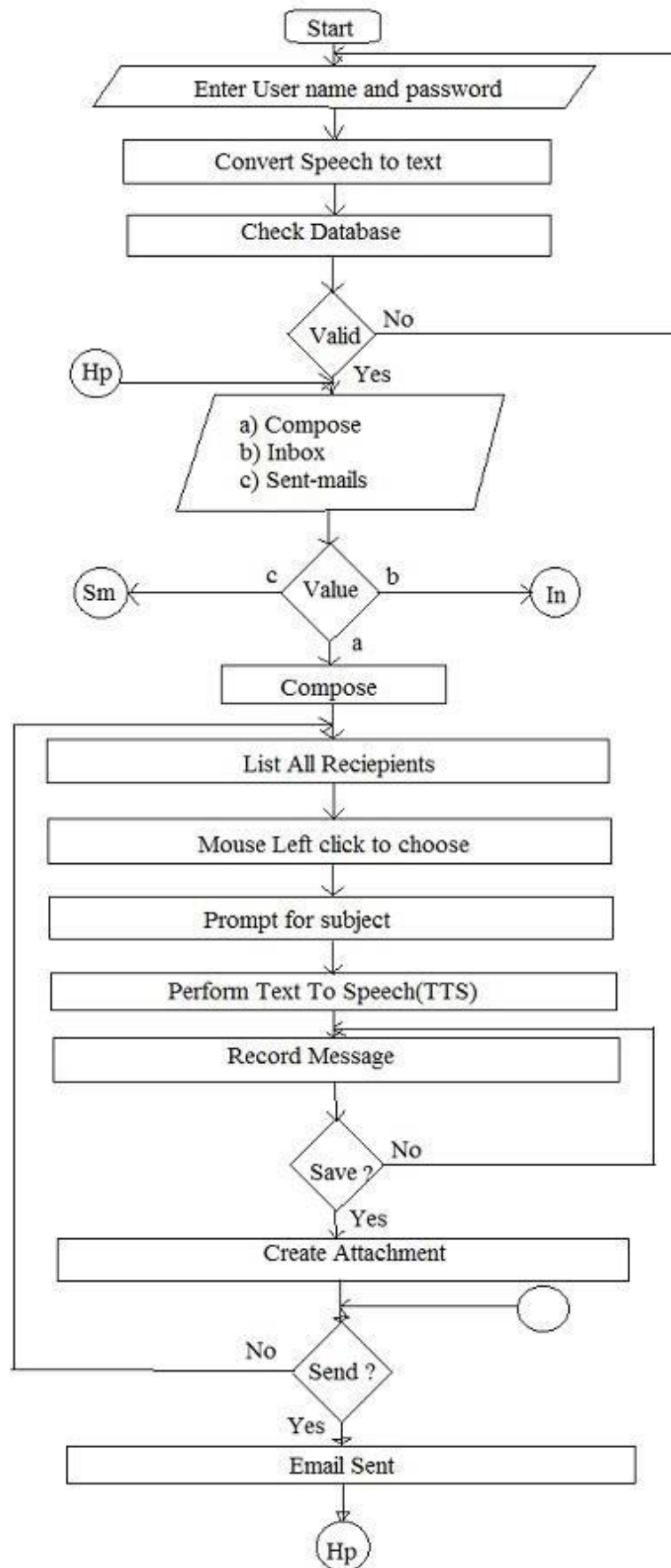
The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

In the design of a system, a number of classes are identified and grouped together in class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses. In order to further describe the behaviour of systems, these class diagrams can be complemented by a state diagram or UML state machine.

### **Activity Diagram**

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modelling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent.

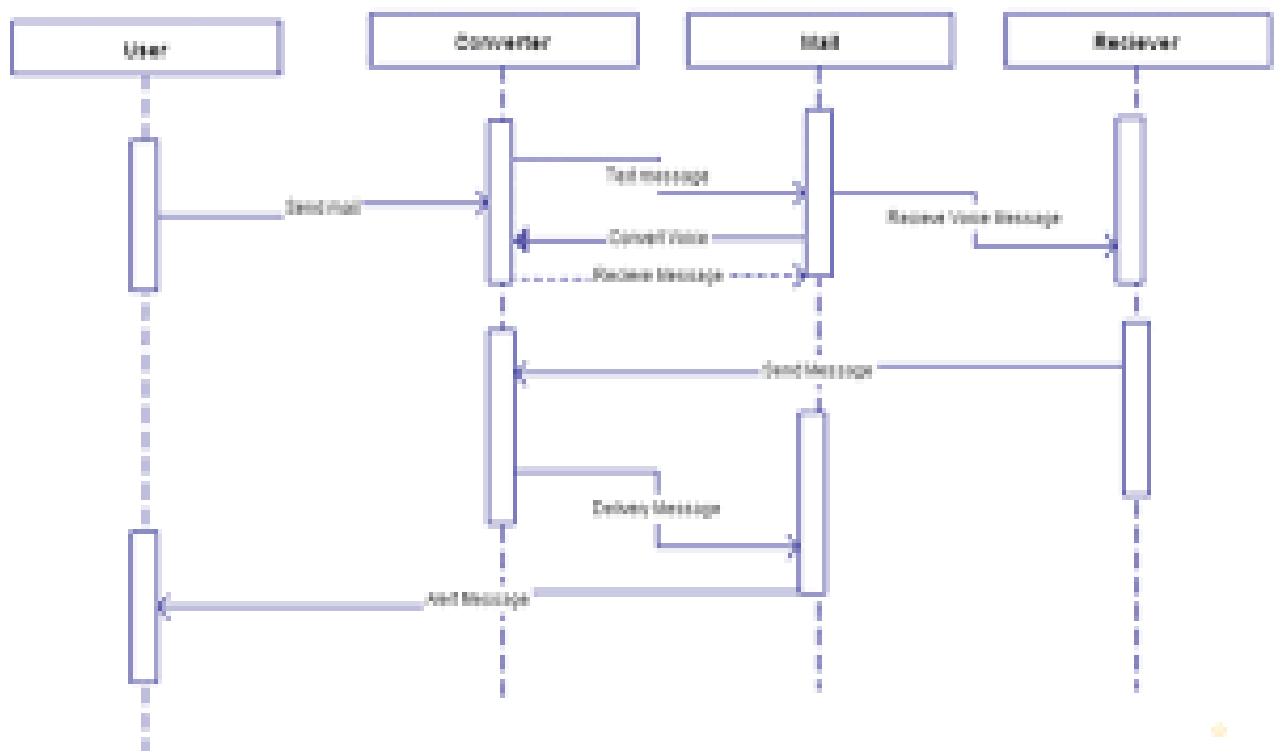
**The Purpose of Activity Diagrams.** The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another. Activity diagrams are constructed from a limited number of shapes, connected with arrows.



## Sequence Diagram

The sequence diagram is a good diagram to use to document a system's requirements and to flush out a system's design. The reason the sequence diagram is so useful is because it shows the interaction logic between the objects in the system in the time order that the interaction takes place.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



## 4 .MODULES DISCRIPTION

### MODULES

- Registration
- Send mail
- User Interface Design
- Reading Unseen mails
- Login

### MODULE DESCRIPTIONS

#### **Registration:**

User has to choose the option Register. We need a trusted second person in order to register because if the blind person pronounciation is not that much clear then the wrong credentials are stored in the database which is not possible to send or read mails in future. For privacy purpose the user has to create a 4-digit keyword which has to be unique the user has to say that keyword in order to access their mail without saying password every time.

The Registration Module holds all the information related to registrations. From within this module, you'll register individuals in courses, make payments and print receipts. You can also cancel and transfer registrations.

#### **User Interface Design:**

Our application user interface is simply a standalone application (Desktop based application) which is very easy to access in a fast way than any other website or app.

User Interface (UI) Design focuses on anticipating what users might need to do and ensuring that the interface has elements that are easy to access, understand, and use to facilitate those actions. UI brings together concepts from interaction design, visual design, and information architecture.

#### **Send mail:**

User has to say the option / send an email as a command. In order to login the user has to say their personal keyword. Then the chatbot asks the receivers email id, Subject, Message. After the chatbot gives a message “Mail Sent Successfully” by voice. Send mail is a server

application that gives businesses a way to send email using the Simple Mail Transfer Protocol (SMTP). It's typically installed on an email server on a dedicated machine that accepts outgoing email messages and then sends these messages to the defined recipient.

### **Reading Unseen mails:**

User has to say the option/ Reading All mails as a command. In order to login the user has to say their personal keyword. Then the chatbot read all the unseen mails and asks the user to give reply or not. The user has to YES/NO as an input Command. **Outlook automatically marks messages as read and removes the blue unread indicator bar as they're opened.** You can also toggle messages as read or unread manually. Select a message in the list. Click the Unread/Read button on the Home tab of the ribbon.

### **Logout Module:**

Logging out means **to end access to a computer system or a website**. Logging out informs the computer or website that the current user wishes to end the login session. Log out is also known as log off, sign off or sign out.



## APPENDIX

### CODING

```
import speech_recognition as sr
import smtplib
# import pyaudio
# import platform
# import sys
from bs4 import BeautifulSoup
import email
import imaplib
from gtts import gTTS
import pygame
import os, time
import pyttsx3

def spe(dum):
    i=0
    print("dum = ",dum)
    tts = gTTS(text=dum, lang='en')
    ttsname = (
        "name.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
    directory. Don't use my directory.
    tts.save(ttsname)

# pygame.lib.load_library('avbin')
# pygame.have_avbin=True

# project: .: Project: Voice based Email for blind :.
# Author: SayakNaskar

# fetch project name
tts = gTTS(text="Project: Voice based Email for blind", lang='en')
ttsname = (
    "name.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
    directory. Don't use my directory.
    tts.save(ttsname)

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

time.sleep(music.duration)
os.remove(ttsname)

# login from os
login = os.getlogin()
print("You are logging from : " + login())

# choices
```

```

print("1. composed a mail.")
tts = gTTS(text="option 1. composed a mail.", lang='en')
ttsname = (
    "hello.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
directory. Don't use my directory.
tts.save(ttsname)

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

time.sleep(music.duration)
os.remove(ttsname)

print("2. Check your inbox")
tts = gTTS(text="option 2. Check your inbox", lang='en')
ttsname = ("second.mp3")
tts.save(ttsname)

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

time.sleep(music.duration)
os.remove(ttsname)

# this is for input choices
tts = gTTS(text="Your choice ", lang='en')
ttsname = (
    "hello.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
directory. Don't use my directory.
tts.save(ttsname)

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

time.sleep(music.duration)
os.remove(ttsname)

# voice recognition part
r = sr.Recognizer()
with sr.Microphone() as source:
    print("Your choices:")
    audio = r.listen(source, timeout=5, phrase_time_limit=10)
    print("ok done!!")

try:
    text = r.recognize_google(audio)
    spe(text)
    print("You said : " + text)

except sr.UnknownValueError:
    print("Google Speech Recognition could not understand audio.")

```

```

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

# choices details
if text == '11' or text == 'One' or text == 'one':
    r = sr.Recognizer() # recognize
    with sr.Microphone() as source:
        print("Your message :")
        audio = r.listen(source, timeout=1, phrase_time_limit=10)
        print("ok done!!")
        try:
            text1 = r.recognize_google(audio)
        except sr.UnknownValueError:
            print("Google Speech Recognition could not understand audio.")
        except sr.RequestError as e:
            print("Could not request results from Google Speech Recognition service; {0}".format(e))

    spe(text1)
    print("You said : " + text1)
    msg = text1
    mail = smtplib.SMTP('smtp.gmail.com', 587) # host and port area
    mail.ehlo() # Hostname to send for this command defaults to the FQDN of the local host.
    mail.starttls() # security connection
    mail.login('classvebbox@gmail.com', 'vebbox@123') # login part
    mail.sendmail('classvebbox@gmail.com', 'shaikharbaas3@gmail.com', msg) # send part
    print("Congrates! Your mail has send. ")
    tts = gTTS(text="Congrates! Your mail has send. ", lang='en')
    ttsname = (
        "send.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
        directory. Don't use my directory.
    tts.save(ttsname)
    music = pygamelet.media.load(ttsname, streaming=False)
    music.play()
    time.sleep(music.duration)
    os.remove(ttsname)
    mail.close()

if text == '2' or text == 'tu' or text == 'two' or text == 'Tu' or text == 'to' or text == 'To':
    mail = imaplib.IMAP4_SSL('imap.gmail.com', 993) # this is host and port area ....ssl
    security
    unm = ('classvebbox@gmail') # username
    psw = ('vebbox@123') # password
    mail.login(unm, psw) # login
    stat, total = mail.select('Inbox') # total number of mails in inbox
    print("Number of mails in your inbox : " + str(total))
    tts = gTTS(text="Total mails are : " + str(total), lang='en') # voice out
    ttsname = (
        "total.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
        directory. Don't use my directory.
    tts.save(ttsname)

```

```

    music = pygamelet.media.load(ttsname, streaming=False)
    music.play()
    time.sleep(music.duration)
    os.remove(ttsname)

    # unseen mails
    unseen = mail.search(None, 'UnSeen') # unseen count
    print("Number of UnSeen mails : " + str(unseen))
    tts = gTTS(text="Your Unseen mail : " + str(unseen), lang='en')
    ttsname = (
        "unseen.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your
    desktop directory. Don't use my directory.
    tts.save(ttsname)
    music = pygamelet.media.load(ttsname, streaming=False)
    music.play()
    time.sleep(music.duration)
    os.remove(ttsname)

    # search mails
    result, data = mail.uid('search', None, "ALL")
    inbox_item_list = data[0].split()
    new = inbox_item_list[-1]
    old = inbox_item_list[0]
    result2, email_data = mail.uid('fetch', new, '(RFC822)') # fetch
    raw_email = email_data[0][1].decode("utf-8") # decode
    email_message = email.message_from_string(raw_email)
    print("From: " + email_message['From'])
    print("Subject: " + str(email_message['Subject']))
    tts = gTTS(text="From: " + email_message['From'] + " And Your subject: " +
        str(email_message['Subject']), lang='en')
    ttsname = (
        "mail.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
    directory. Don't use my directory.
    tts.save(ttsname)
    music = pygamelet.media.load(ttsname, streaming=False)
    music.play()
    time.sleep(music.duration)
    os.remove(ttsname)

    # Body part of mails
    stat, total1 = mail.select('Inbox')
    stat, data1 = mail.fetch(total1[0], "(UID BODY[TEXT])")
    msg = data1[0][1]
    soup = BeautifulSoup(msg, "html.parser")
    txt = soup.get_text()
    print("Body : " + txt)
    tts = gTTS(text="Body: " + txt, lang='en')
    ttsname = (
        "body.mp3") # Example: path -> C:\Users\sayak\Desktop> just change with your desktop
    directory. Don't use my directory.
    tts.save(ttsname)

```

```

    music = pygamelet.media.load(ttsname, streaming=False)
    music.play()
    time.sleep(music.duration)
    os.remove(ttsname)
    mail.close()
    mail.logout()

from tkinter import *
global main_screen
from tkinter import Tk, font, messagebox
from tkcalendar import Calendar, DateEntry
global root,password,username,password_entry,username_entry,idma
import spe
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="voice_email"
)
mycursor = mydb.cursor()
def log_again():
    pass
    spe.spe("one to login-in again ,two to quit ")
    log_again_val = spe.tex()
    if log_again_val==1:
        test()
    else:
        quit()

def test():
    global idma
    spe.spe("Tell me your personal id to login")
    id = spe.tex()
    idma=id
    e.insert(END, id)
    spe.spe(id)
    print(id)

sql = "select name from register_user where id = %s; "
mycursor.execute(sql,[id])
    results = mycursor.fetchone()
    # print(len(results))
    if len(results) is not None:
        speak_res=f"Welcome to voice based email {results}"
        spe.spe(speak_res)
        spe.spe("logged-in Successfully ")
        voice_mod()
    else:
        spe.spe("check your loggin Id ,LogginUnsuccessfull ")

```

```
log_again()
```

```
def login_by_voice():
    global e, root1
    main_screen.destroy()
    root1 = Tk()
    root1.geometry('600x300')
    root1.title("User voice login Form")
    Label(root1, text="LOGIN By VOICE-PORTAL ", bg="#4169e1", width="300", height="3",
           font=("Times New Roman", 18, 'bold', 'italic')).pack()
    Label(text="").pack()
```

```
voice_id = Label(root1, text="Login ID", bg="#4169e1", width=20,
                   font=("Times New Roman", 16, 'bold', 'italic'))
voice_id.place(x=10, y=130)
e = Entry(root1, width=22,
          bg='#89c8ff', font=("Times New Roman", 16, 'bold', 'italic'))
e.place(x='280', y='130', height=30, width=300)
```

```
test()
# voice_mod()
```

```
def voice_mod():
    global vroot1
```

```
spe.spe("1 for open mail")
spe.spe(" 2 for view unopened mail")
spe.spe(" 3 for view All mail")
spe.spe(" 4 for logout or close")
id = spe.tex()
print(id)
print(type(id))
```

```
def vopen():
    # root1.destroy()
    vop = Tk()
    vop.geometry('700x600')
    vop.title("Mail open by voice")
    Label(vop, text="Voice Message Compose Page", bg="#4169e1", width="300", height="3",
           font=("Times New Roman", 16, 'bold', 'italic')).pack()
    Label(text="").pack()
    Label(vop, width=20, height=1, text="From", bg='#89c8ff', font=("Times New Roman", 16,
        'bold', 'italic')).place(
        x=20, y=100)

    fromtex = Entry(vop, width=30, bg='#89c8ff'
                    , font=("Times New Roman", 16, 'bold', 'italic'))
```

```

fromtex.place(x=320, y=100)

Label(vop, text="Subject", width=20, height=1, bg='#89c8ff'
      , font=("Times New Roman", 16, 'bold', 'italic')).place(x=20, y=200)
content = Entry(vop, width=30,
bg='#89c8ff', font=("Times New Roman", 16, 'bold', 'italic'))
content.place(x=320, y=200)
Label(vop, text="TO ", width=20, height=1, bg='#89c8ff'
      , font=("Times New Roman", 16, 'bold', 'italic')).place(x=20, y=300)

totex = Entry(vop, width=30,
bg='#89c8ff', font=("Times New Roman", 16, 'bold', 'italic'))
totex.place(x=320, y=300)

spe.spe(" please tell the from adress or login id")
fromid = spe.tex()
spe.spe(fromid)
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="voice_email"
)
mycursor = mydb.cursor()
mycursor = mydb.cursor()
sql = f"select name from register_user where id = %s; "
mycursor.execute(sql, [(fromid)])
results = mycursor.fetchone()
print(results)
if results is not None:
spe.spe(results)
spe.spe(" please tell the message ")
msgid = spe.tex()
spe.spe(msgid)
content.insert(END, msgid)

spe.spe(" please tell To adress or login id")
toid = spe.tex()
spe.spe(toid)
totex.insert(END, toid)

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="voice_email"
)
mycursor = mydb.cursor()
sql = "select name from register_user where id = %s; "
mycursor.execute(sql, [(toid)])
results = mycursor.fetchone()

```

```

        if results is not None:
sendnamespe = f"sender name is {results[0]}"
spe.spe(sendnamespe)
        mycursor1 = mydb.cursor()
        sql1 = "insert into compose (fromid,subject,toid,readmsg) values(%s,%s,%s,%s); "
val = (fromid, msgid, toid, '0')
        mycursor1.execute(sql1, val)
mydb.commit()
messagebox.showinfo("showinfo", "Message send Sucessfully")
vop.mainloop()
        else:
messagebox.showinfo("showinfo", "wrong id")
        else:
messagebox.showinfo("showinfo", "id not found or not registered")
        # fromtex.insert(END, fromid)

def vview():
    t = 0
    # root1.destroy()
vvi = Tk()
vvi.geometry('1000x600')
vvi.title("View Mail open by voice")
spe.spe("welcome to inbox message reading")
    con = mysql.connector.connect(host="localhost", user="root", password="",
database="voice_email")
    cur = con.cursor()
    print('idma=',idma)
cur.execute(f"SELECT * from compose where toid={idma} and readmsg='0';")
    res = cur.fetchall()
    # print(res)
    if len(res) == 0:
Label(vvi, text="All Message has been readed", bg="#4169e1", width="300", height="3",
        font=("Times New Roman", 16, 'bold', 'italic')).pack()
        Label(text="").pack(pady=20, padx=10)
    else:
i = 1
        for n in res:
            for j in range(len(n)):
                e = Label(vvi, width=25, fg="blue", text=n[j], borderwidth=2, relief="ridge",
anchor="w",
padx=5,
pady=8)
e.grid(row=i, column=j)
                e = Label(vvi, width=25, fg="#000", text="id", borderwidth=2, relief="ridge",
anchor="w",
padx=10, pady=8)
e.grid(row=0, column=0)
                e = Label(vvi, width=25, fg="#000", text="fromid", borderwidth=2, relief="ridge",
anchor="w",
padx=10, pady=8)
e.grid(row=0, column=1)

```



```

        e = Label(vvi, width=25, fg="#000", text="subject", borderwidth=2,
relief="ridge",
                    anchor="w",
padx=10, pady=8)
e.grid(row=0, column=2)
        e = Label(vvi, width=25, fg="#000", text="toid", borderwidth=2, relief="ridge",
                    anchor="w",
padx=10, pady=8)
e.grid(row=0, column=3)
        e = Label(vvi, width=25, fg="#000", text="readmsg", borderwidth=2,
relief="ridge",
                    anchor="w",
padx=10, pady=8)
e.grid(row=0, column=4)

        # e.insert(END,res[j])
i = i + 1

mycursor = mydb.cursor()

mycursor.execute(f"SELECT fromid,subject FROM compose where toid={idma} and
readmsg='0'; ")
myresult = mycursor.fetchall()

        if len(myresult) >= 1:
            for x, y in myresult:
                rems = f"From ID is {x}"
spe.spe(rems)
                rems1 = f"Subject is {y} "
spe.spe(rems1)
sql = f"UPDATE compose SET readmsg = '1' WHERE fromid = {x};"
mycursor.execute(sql)
mydb.commit()
                t = t + 1
readcount = f".....,{t} message readed "
spe.spe(readcount)
                # print(mycursor.rowcount)
            else:
spe.spe("TILL now ALL the message has been readedd")
vvi.mainloop()

def veiwall():
    # root1.destroy()
    t = 0
viall = Tk()
viall.geometry('1000x600')
viall.title("Veiw all Mail open by voice")
    con = mysql.connector.connect(host="localhost", user="root", password="",
database="voice_email")
    cur = con.cursor()
cur.execute(f"SELECT * from compose where fromid={idma} ")

```

```

        # print(id)
        res = cur.fetchall()
        print(res)
        if len(res) == 0:
            Label(viall, text="All Message has been readed", bg="#4169e1", width="300", height="3",
                  font=("Times New Roman", 16, 'bold', 'italic')).pack()
            Label(text="").pack(pady=20, padx=10)

        else:
            i = 1
            for n in res:
                for j in range(len(n)):
                    e = Label(viall, width=25, fg="blue", text=n[j], borderwidth=2, relief="ridge",
                              anchor="w",
                              padx=5,
                              pady=8)
                    e.grid(row=i, column=j)
                    e = Label(viall, width=25, fg="#000", text="id", borderwidth=2, relief="ridge",
                              anchor="w",
                              padx=10, pady=8)
                    e.grid(row=0, column=0)
                    e = Label(viall, width=25, fg="#000", text="fromid", borderwidth=2,
                              relief="ridge",
                              anchor="w",
                              padx=10, pady=8)
                    e.grid(row=0, column=1)
                    e = Label(viall, width=25, fg="#000", text="subject", borderwidth=2,
                              relief="ridge",
                              anchor="w",
                              padx=10, pady=8)
                    e.grid(row=0, column=2)
                    e = Label(viall, width=25, fg="#000", text="toid", borderwidth=2, relief="ridge",
                              anchor="w",
                              padx=10, pady=8)
                    e.grid(row=0, column=3)
                    e = Label(viall, width=25, fg="#000", text="readmsg", borderwidth=2,
                              relief="ridge",
                              anchor="w",
                              padx=10, pady=8)
                    e.grid(row=0, column=4)

                    # e.insert(END,res[j])

            i = i + 1

        if len(res) >= 1:
            for a, x, y, z, b in res:
                rems = f"From ID is {x}"
                spe.spe(rems)
                rems1 = f"Subject is {y} "

```

```

spe.spe(rem1)
    rem2 = f"ToooI d is {z} "
spe.spe(rem2)
sql = f"UPDATE compose SET readmsg = '1' WHERE fromid = {x};"
cur.execute(sql)
mydb.commit()
    t = t + 1
readcount = f".....,{t} message readed "
spe.spe(readcount)
    # print(mycursor.rowcount)
else:
spe.spe("TILL now ALL the message has been readedd")
viall.mainloop()

def close():
spe.spe("closing the app")
    # root1.destroy()
quit()

if id=="one" or id=="ONE " or id=='1' :

vopen()
voice_mod()

elif id== "two" or id== "tu" or id== "TWO" or id=='2' :

vview()
voice_mod()
elif id=="three" or id=="THREE" or id=="tree" or id== '3' or id=='free' or id=='cri':

veiwall()
voice_mod()
elif id=="FOUR" or id=="four" or id=="for" or id=='4' :
spe.spe("closing application")
close()
else:
spe.spe("give the correct command")
voice_mod()

vview()
def admin_view_user():
ad_view_user=Tk()
ad_view_user.title("Register")
ad_view_user.geometry("800x650")
    con = mysql.connector.connect(host="localhost", user="root", password="",
database="voice_email")
    cur = con.cursor()
cur.execute(f"SELECT * from register_user")
    res = cur.fetchall()

```

```

i = 0
for n in res:
    for j in range(len(n)):
        e = Label(ad_view_user, width=15, fg="blue", text=n[j], borderwidth=2, relief="ridge",
anchor="w", padx=10,
pady=8)
e.grid(row=i, column=j)
        e = Label(ad_view_user, width=15, fg="#000", text="name", borderwidth=2,
relief="ridge", anchor="w",
padx=10, pady=8)
e.grid(row=0, column=0)
        e = Label(ad_view_user, width=15, fg="#000", text="date", borderwidth=2,
relief="ridge", anchor="w",
padx=10, pady=8)
e.grid(row=0, column=1)
        e = Label(ad_view_user, width=15, fg="#000", text="finger", borderwidth=2,
relief="ridge", anchor="w",
padx=10, pady=8)
e.grid(row=0, column=2)
        e = Label(ad_view_user, width=15, fg="#000", text="message", borderwidth=2,
relief="ridge", anchor="w",
padx=10, pady=8)
e.grid(row=0, column=3)

        # e.insert(END,res[j])
i = i + 1

def admin_reg_user():
    global fname,mail,cal

    root = Tk()
    root.geometry('600x500')
    root.title("User Registration Form")
    fullname=StringVar()
    email=StringVar()

    Label(root, text="Admin Register & view Page", bg="#4169e1", width="300", height="3",
font=("Times New Roman", 16, 'bold', 'italic')).pack()
    Label(text="").pack(pady=20, padx=10)

    label_1 = Label(root, text="FullName", bg="#4169e1",width=20, font=("Times New
Roman", 16, 'bold', 'italic'))
    label_1.place(x=10, y=130)

    fname = Entry(root, textvariable=fullname, width=22, bg='#89c8ff'
, font=("Times New Roman", 16, 'bold', 'italic'))
    fname.place(x=280, y=130,height=30,width=300)

    label_2 = Label(root, text="Email", bg="#4169e1",width=20, font=("Times New Roman",
16, 'bold', 'italic'))
    label_2.place(x=10, y=180)

```

```

mail = Entry(root, textvariable=email, width=22, bg='#89c8ff'
            , font=("Times New Roman", 16, 'bold', 'italic'))
mail.place(x=280, y=180, height=30, width=300)

label_3 = Label(root, text="Gender", bg="#4169e1",width=20, font=("Times New Roman",
16, 'bold', 'italic'))
label_3.place(x=10, y=230)
var = IntVar()
gen=Radiobutton(root, text="Male", variable=var,width=10 ,font=("Times New Roman",
14, 'bold', 'italic'), value=1)
gen.place(x=255, y=230)
gen1=Radiobutton(root, text="Female",width=10, font=("Times New Roman", 14, 'bold',
'italic'), variable=var, value=2)
gen1.place(x=380, y=230)

label_4 = Label(root, text="Age:", bg="#4169e1",width=20, font=("Times New Roman",
16, 'bold', 'italic'))
label_4.place(x=10, y=280)

cal = DateEntry(root, width=30, background="#89c8ff", foreground="#89c8ff", bd=2)
cal.place(x=280, y=280, height=30, width=300)

# *****-----insert valuesin MYSQL database-----
*****

def user_login():
print("fname = ", fname.get())
print("email = ", mail.get())
print("date = ", cal.get())

us_lo_name=fname.get()
us_lo_email=mail.get()
us_lo_date=cal.get()
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="voice_email"
)
mycursor = mydb.cursor()
sql = "INSERT INTO register_user (name,email,DOB) VALUES (%s, %s,%s)"
val = (us_lo_name,us_lo_email,us_lo_date)
mycursor.execute(sql, val)
mydb.commit()
    # print(mycursor.rowcount, "record inserted.")
messagebox.showinfo("showinfo", "Registered Sucessfully")
    # adminv1.destroy()

# *****-----insert valuesin MYSQL database finishes-----

```

\*\*\*\*\*

```
root.destroy()
```

```
    # login_by_voice()
```

```
Button(root, text='Submit', width=20, bg="#4169e1",command=user_login, font=("Times New Roman", 16, 'bold', 'italic'), fg='white').place(x=190, y=340)
```

```
root.mainloop()
```

```
print("registration form seccussfully created...")
```

```
def admin_veiw():
```

```
    global adminv1
```

```
register_screen.destroy()
```

```
    adminv1 = Tk()
```

```
    adminv1.title("Register")
```

```
    adminv1.geometry("500x450")
```

```
    # Label(adminv1,text="Admin Register & view Page", bg="blue", width="300", height="2", font=("Calibri", 13)).pack()
```

```
    # Label(text="").pack()
```

```
Label(adminv1, text="Admin Register & view Page", bg="#4169e1", width="300", height="3",
```

```
    font=("Times New Roman", 16, 'bold', 'italic')).pack()
```

```
    Label(text="").pack(pady=20, padx=10)
```

```
    # create Login Button
```

```
Button(adminv1,text="Register user", bg="#4169e1", width="30", height="2",
```

```
    font=("Times New Roman", 16, 'bold', 'italic'),
```

```
command=admin_reg_user).pack(padx=10,pady=20)
```

```
    Label(text="").pack()
```

```
    # create a register button
```

```
Button(adminv1,text="Veiw User",bg="#4169e1", width="30", height="2",
```

```
    font=("Times New Roman", 16, 'bold',
```

```
'italic'),command=admin_view_user).pack(padx=10,pady=30)
```

```
    Label(text="").pack()
```

```
    adminv1.mainloop()
```

```
def admin_login():
```

```
    global main_screen,password,username,username_entry,password_entry,register_screen
```

```
    # main_screen.dest
```

```
main_screen.destroy()
```

```
register_screen = Tk()
```

```
register_screen.title("Admin Login")
```

```
register_screen.geometry("500x450")
```

```
    # Set text variables
```

```

username = StringVar()
password = StringVar()

# Set label for user's instruction
Label(register_screen,text="Admin registration", bg="#4169e1", width="300", height="3",
      font=("Times New Roman", 16, 'bold', 'italic')).pack()
Label(text="").pack(pady=20,padx=10)

# Set username label
Label(register_screen,width=20,height=1, text="Username * ",
      bg='#89c8ff',font=("Times New Roman", 16,'bold','italic')).pack(padx=10,pady=20)

# Set username entry
# The Entry widget is a standard Tkinter widget used to enter or display a single line of text.

username_entry = Entry(register_screen, textvariable=username,width=22,bg='#89c8ff'
      ,font=("Times New Roman", 16,'bold','italic'))
username_entry.pack(padx=10,pady=10)

# Set password label
Label(register_screen, text="Password * ",width=20,height=1,bg='#89c8ff'
      ,font=("Times New Roman", 16,'bold','italic')).pack(padx=10,pady=10)

# Set password entry
password_entry = Entry(register_screen, textvariable=password, show='*',width=22,
      bg='#89c8ff',font=("Times New Roman", 16,'bold','italic'))
password_entry.pack(padx=10,pady=10)

Label(register_screen, text="").pack()

def value():
    print("pass = ", password_entry.get())
    print("username = ", username_entry.get())
    aname=password_entry.get()
    auser=username_entry.get()
    if aname=="admin" and auser=="admin" :
        admin_veiw()
    else:
        messagebox.showinfo("showinfo", "Login Failed")
    quit()

# Set register button
Button(register_screen, text="Register", width=10, height=1,
      bg="#4169e1",font=("Times New Roman", 16,'bold','italic'), command=value).pack()
# main_screen.mainloop()

def main_account_screen():
    global main_screen
    main_screen = Tk() # create a GUI window
    main_screen.geometry("500x450") # set the configuration of GUI window

```

```

main_screen.title("Account Login") # set the title of GUI window

# create a Form label
Label(text="Choose Login Or Register", bg="#4169e1", width="300", height="3",
font=("Times New Roman", 16,'bold','italic')).pack()
Label(text="").pack()

# create Login Button
Button(text="Admin", height="2", width="30",bg='#4169e1',font=("Times New Roman",
16,'bold','italic'),command=admin_login).pack()
Label(text="").pack()

# create a register button
Button(text="User", height="2", width="30",bg='#4169e1',font=("Times New Roman",
16,'bold','italic'),command=login_by_voice).pack()

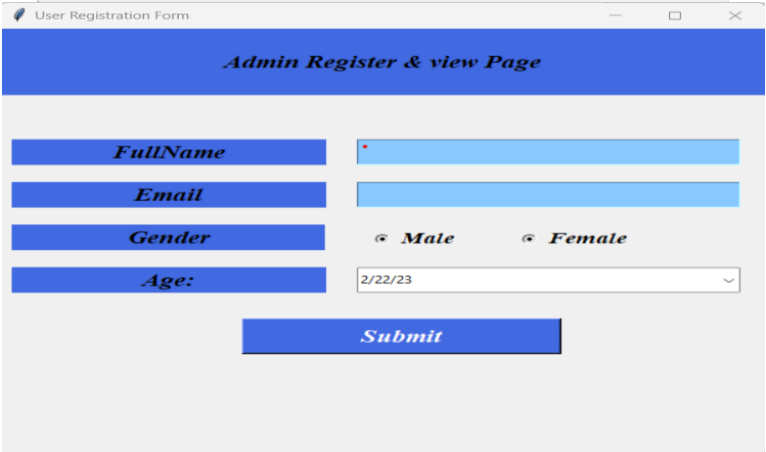
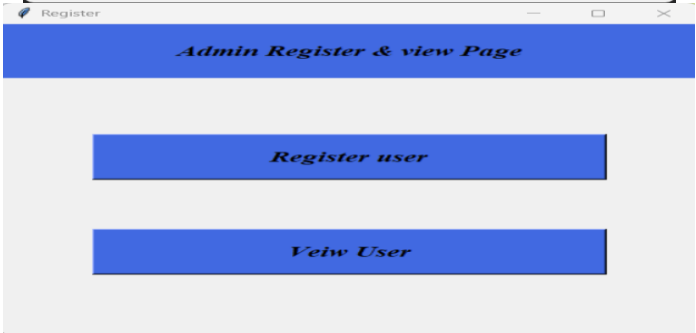
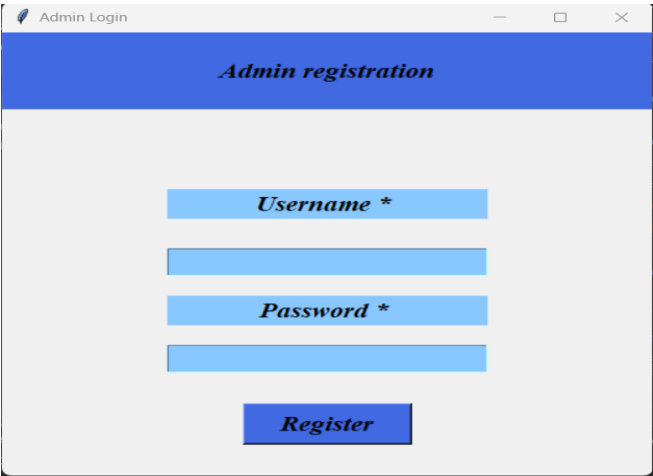
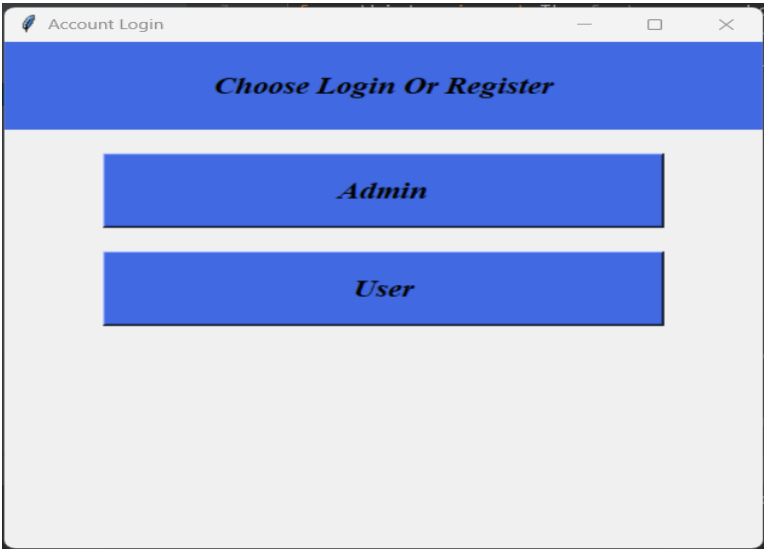
main_screen.mainloop() # start the GUI

main_account_screen()

```



# SCREENSHOTS



Register			
name	date	finger	message
1002	df	df	df
1003	shaikh	sha@gmail.com	10/10/10
1004	divya	divyakavya2529@gmail	11/28/98
1005	ddd	ddd	5/7/22

User voice login Form

**LOGIN By VOICE-PORTAL**

**Login ID**

Mail open by voice

**Voice Message Compose Page**

**From**

**Subject**

**TO**

View Mail open by voice

**All Message has been readed**

## **5.SYSTEMSPECIFICATION**

### **5.1HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation the system and should there for e be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design

- Processor : Dual core processor2.6.0 GHZ
- RAM : 2GB
- HARD DISK : 160 GB
- Compact Disk : 650 Mb
- Keyboard : Standard Keyboard
- Monitor : 15 inch Color monitor

### **5.2 SOFTWARE REQUIREMENTS:**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is useful in estimating cost, planningteamactivitiesandperformingtasksthroughoutthedevelopmentactivity.

- Operating system : Windows OS (7,8,8.1,10)
- Front-End : Tkinter
- Back-End : PYTHON, MYSQL
- LANGUAGE : PYTHON
- Server : XAMP

### 5.3 SOFTWARE DESCRIPTION:

#### **Python:**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985-1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

#### **Audience:**

This tutorial is designed for software programmers who need to learn Python programming language from scratch.

- **Python is Interpreted** – Python is processed at run time by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

#### **History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

## Python Features

Python's features include—

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read**—Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain**—Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows ,and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** –You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases**—Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apartfromtheabove-

mentionedfeatures,Pythonhasabiglistofgoodfeatures,fewarelistedbelow—

- It supports functional and structured programming methods as well as OOP.
- Itcanbeusedasascriptinglanguageorcanbecompiledtobyte-codeforbuildinglargeapplications.
- Itprovidesveryhigh-leveldynamicdatatypesandsupportsdynamictypechecking.
- It supports automatic garbage collection.

- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OSX. Let's understand how to setup our Python environment.

### **Local Environment Setup**

Open a terminal window and type "python" to find out if it is already installed and which version is installed.

- Unix(Solaris, Linux ,FreeBSD, AIX,HP/UX ,Sun OS, IRIX, etc.)
- Win9x/NT/2000
- Macintosh(Intel, PPC,68K)
- OS/2
- DOS(multiple versions)
- Palm OS
- Nokia mobile phones
- Windows CE
- Acorn/RISCOS
- BeOS
- Amiga
- VMS/OpenVMS
- QNX
- V x Works
- Psion
- Python has also been ported to the Java and .NET virtual machines

### **Getting Python**

The most up-to-date and current source code, binaries, documentation, news, etc. is available on the official website of Python <https://www.python.org/>

You can download Python documentation from <https://www.python.org/doc/>.

The documentation is available in HTML, PDF, and Post Script formats.

## Installing Python

Python distribution is available for a wide variety of platforms. You need to download only the binary code applicable for your platform and install Python.

If the binary code for your platform is not available, you need a C compiler to compile the source code manually. Compiling the source code offers more flexibility in terms of choice of features that you require in your installation.

Here is a quick over view of installing Python on various platforms—

### Unix and Linux Installation

Here are the simple steps to install Python on Unix/Linux machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.
- Follow the link to download zipped source code available for Unix/Linux.
- Download and extract files.
- Editing the *Modules/Set up* file if you want to customize some options.
- run `./configure` script
- `make`
- `make install`

This installs Python at standard location `/usr/local/bin` and its libraries at `/usr/local/lib/python XX` where XX is the version of Python.

### Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.
- Follow the link for the Windows installer *python-XYZ.msi* file where XYZ is the version you need to install.
- To use this installer *python-XYZ.msi*, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

## Mac into sh Installation

Recent Macs come with Python installed, but it may be several years out of date. See <http://www.python.org/download/mac/> for instructions on getting the current version along with extra tools to support development on the Mac. For older Mac OS's before Mac OS X10.3(released in 2003), Mac Python is available.

Jack Jansen maintains it and you can have full access to the entire documentation [this website—  
http://www.cwi.nl/~jack/macpython.html](http://www.cwi.nl/~jack/macpython.html). You can find complete installation details for Mac OS installation.

## Setting up PATH

Programs and other executable files can be in many directories, so operating systems provide a search path that lists the directories that the OS searches for executables.

The path is stored in an environment variable, which is a named string maintained by the operating system. This variable contains information available to the command shell and other programs.

The **path** variable is named as PATH in Unix or Path in Windows (Unix is case sensitive; Windows is not).

In Mac OS, the installer handles the path details. To invoke the Python interpreter from any particular directory, you must add the Python directory to your path.

Setting path that Unix/Linux

To add the Python directory to the path for a particular session in Unix—

- **In the csh shell**—type `setenv PATH "$PATH:/usr/local/bin/python"` and press Enter.
- **In the bash shell (Linux)**—type `export PATH="$PATH:/usr/local/bin/python"` and press Enter.
- **In the sh or ksh shell**—type `PATH="$PATH:/usr/local/bin/python"` and press Enter.
- **Note** —/usr/local/bin/python is the path of the Python directory



## Setting path that Windows

To add the Python directory to the path for a particular session in Windows

**At the command prompt** –type path% path%;C:\Pythonand press Enter.

**Note** –C:\Pythonis the path of the Python directory

## Python Environment Variables

Here are important environment variables, which can be recognized by Python

## MYSQL INTRODUCTION

The My SQL® data base has become the world's most popular open source data base because of its consistent fast performance, high reliability and ease of use. It's used on every continent--Yes, even Antarctica! --by individual Web developers as well as many of the world's largest and fastest-growing organizations to save time and money powering their high-volume Web sites, business-critical systems and packaged software -- including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, and Zappos.com.

Not only is My SQL the world's most popular open source database, it's also become the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, My SQL, PHP / Perl / Python.) My SQL runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris, HP-UX, IBM AIX, giving you the kind of flexibility that puts you in control.

Whether you're new to database technology or an experienced developer or DBA, My SQL offers a comprehensive range of certified software, support, training and consulting to make you successful.

My SQL can be built and installed manually from source code, but this can be tedious so it is more commonly installed from a binary package unless special customizations are required. On most Linux distributions the package management system can download and install MySQL with minimal effort, though further configuration is often required to adjust security and optimization settings.

Though My SQL began as a low-end alternative to more powerful proprietary databases, it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server deployments, either as a component in a LAMP based web application or as a standalone data base server. Much

of My SQL's appeal originates in its relative simplicity and ease of use, which is enabled by an ecosystem of open source tools such as php My Admin. In the medium range ,My SQL can be scaled by deploying it on more powerful hardware, such as a multi-processor server with gigabytes of memory.

There are however limits to how far performance can scale on a single server, so on larger scales, multi-server My SQL deployments are required to provide improved performance and reliability. A typical high-end configuration can include a powerful master database which handles data write operations and is replicated to multiple slaves that handle all read operations.[18] The master server synchronizes continually with its slaves so in the event of failure a slave can be promoted to become the new master, minimizing downtime. Further improvements in performance can be achieved by caching the results from database queries in memory using memcached, or breaking down a database into smaller chunks called shards which can be spread across a number of distributed server clusters.

## **5.4 HARDWARE DESCRIPTION**

### **Camera:**

An Internet Protocol camera, or IP camera, is a type of digital video camera that receives control data and sends image data via the Internet. They are commonly used for surveillance. Unlike analog closed-circuit television (CCTV) cameras, they require no local recording device, but only a local area network. Most IP cameras are webcams, but the term IP camera or net cam usually applies only to those used for surveillance that can be directly accessed over a network connection.

, video and alarm management. Others are able to operate in a decentralized manner with no NVR needed, as the camera is able to record directly to any local or remote storage media. The first centralized IP camera was Axis Net eye 200, released in 1996 by Axis Communications. WIFI Home camera is the IP cameras that are sitting in the homes, and that utilize high speed broadband Internet access to capture and transmit High-definition video.[11] The gap of IP camera used in small business and home is diminishing. Current home security camera can be used in small business due to its easy installation, that translates no installers required so it's far more cost and time savings. The exception is to monitor large businesses or commercial space like a mall. Some will

require super high resolution videos (ie 4K) and will a great many cameras(possibly up to hundreds) and professional applications to accommodate these type of needs.



One of most popular applications is the Wifi home security cameras allows users to remotely view their homes indoors and outdoors via an Mobile app installed on their Mobile device .Most cameras offer such features as wide angle (around 140 degree, or pan/tilt up to 350 degrees horizontal, 90 degree vertical), low/night vision, and motion activation notifications. When an event occurs, users will receive alarms via an app. Video clips are stored in the local device (if a Micro-SD is present) or on the Cloud computing

The market size of home security system has reached \$4.8 billion dollars in 2018. This is displayed a CAGR of 22.4% between 2011 and 2018. Countries such as including Uganda, Sweden, Italy, and Tanzania suffered from high crime rates particularly robbery and theft, are keen to adopting the home security cameras. In addition, two countries, US and China, have high implementation rate of residential security cameras. Major key players in home security market are Nest (US), Ring (owned by Amazon, US), Arlo (owned by Net gear, US), and SimpliSafe (US). Hikvision Digital Technology (Ltd) and Leshi Video Tech (China) are the largest camera manufacturers. As for alarm security industry, key players are ADT Inc., Security Services (US), Vivint Inc, and Front Point Security Solutions.

### **5.5 TESTING PROCEDURE**

A test case is an asset of data that the system will process as normal input. The strategies that we have used in our project are,

#### **SYSTEM TESTING**

Testing is the stage of implementation of which aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct the goal will be achieved. The candidate's system is subjected to a variety of tests. Online response, volume, stress, recovery, security and usability tests. A series of testing are performed for the proposed system before the system is ready for user acceptance testing.

#### **UNIT TESTING**

The procedure level testing is made first. By giving improper inputs, the errors occur, are noted and eliminated. Then the web form level is made.

#### **INTEGRATION TESTING:**

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions.

Thus the system testing is a confirmation that all its correct and an opportunity to show the user that the system works.

### **VALIDATION TESTING:**

The final step involves validation testing which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called “Alpha and Beta test” to uncover that only the end user seems able to find. The compilation of the entire project is based on the full satisfaction of then d users.

### **ACCEPTANCE TESTING:**

Acceptance testing can be defined in many ways, but a simple definition is the succeeds when the software functions in a manner that can be reasonable expected by the customer. After the acceptance test has been conducted, one of the two possible conditions exists. This is to fine whether the inputs are accepted by the database or other validations. For example accept only numbers in the numeric field, date format data inthe date field. Also the null check for the not null fields. If any error occurs then showthe error messages. The function of performance characteristics to specification and is accepted. A deviation from specification is uncovered and a deficiency list is created.

### **WHITEBOX TESTING**

White box testing, sometimes called "Glass-box testing". Using white box testing methods, the following tests were made on the system,

- All in dependent paths with in a module have been exercised at least once
- All logical decisions were checked for the true and false side of the values.
- All loops were executed to check their boundary values.
- Internal data-structure was tested for their validity.

## **BLACKBOX TESTING**

Black box testing focuses on the functional requirements of the software. That is black box testing enables the software engineer to drive a set of input conditions that willfully exercise the requirements for a program. Black box testing is not an alternative for white box testing techniques. Rather, it is a complementary approach that is likely to uncover different class of errors .Black box testing attempts to find errors in the following categories:

- Interface errors.
- Performances in data structures or external data base access.
- Performance errors.
- Initialization and termination errors.

Incorrect or missing functions.

## **6.CONCLUSION**

Our application is a user-friendly, efficient and economical system, which allows a physically challenged individual to interact with an Python application easily. It involves the development and implementation of a real-time email interaction system for the visually impaired. We have planned to develop a system that allows visually impaired individuals to access email services in an efficient way. Our application can help in overcoming some of the drawbacks of the existing email systems. In this system, the use of the keyboard has been eliminated completely. This reduces the cognitive load of remembering keyboard shortcuts as well as the position of the keys on a keyboard. The user only requires listening to the voice commands given by the system and responding accordingly in order to get the desired operations performed. This requires the user to speak the operation in the email application and then the system will perform the necessary operations. If necessary, the user would provide information through voice inputs, and the system would ensure that the user's details are authenticated.

## 7.REFERENCE

- [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] Khan, R., Sharma, P. K., Raj, S., Verma, S. K., &Katiyar, S. Voice-Based E-Mail System using Artificial Intelligence.
- [3] Pathan, N., Bhoyar, N., Lakra, U., &Lilhare, D. (2019). V-Mail (Voice Based E-Mail Application).
- [4] Sawant, S., Wani, A., Sagar, S., Vanjari, R., &Dhage, M. R. (2018). Speech Based E-mail System for Blind and Illiterate People.
- [5] Jayachandran, K., &Anbumani, P. (2017). Voice-based email for blind people.
- [6] Ingle, P., Kanade, H., &Lanke, A. (2016). Voice-based e-mail System for Blinds.
- [7] Runze Chen, Zhanhong Tian, Hailun Liu, Fang Zhao, Shuai Zhang, Haobo Liu "Construction of a Voice-Driven Life Assistant System for Visually Impaired People" (2018).
- [8]. Jagtap Nilesh, Pawan Alai, Chavhan Swapnil and Bendre M.R. , "Voice Based System in Desktop and Mobile Devices for Blind People ". International Journal of Emerging Technology and Advanced Engineering.
- [9]. K. Jayachandran, P. Anbumani, "Voice Based Email for Blind People ", IJARIT, 2017.
- [10]. Pranjal Ingle, HarshadaKanade, Arti Lanke, "Voice based e-mail System for Blinds ", IJRSCSE,2016.