VOICE BASED EMAIL SYSTEM FOR BLINDS

A Project Report Submitted by

Diksha Kumari
(4NM13CS057)

Nisha Pai
(4NM13CS101)

Parul Jha
(4NM13CS110)

Pratmika Nayak
(4NM13CS125)

UNDER THE GUIDANCE OF

Mr. VIJAYA MURARI T

Assistant Professor

Dept of Computer Science and Engineering

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

Bachelor of Engineering in
Computer Science & Engineering

from

Visvesvaraya Technological University, Belgaum

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum) (AICTE approved, NBA Accredited, ISO 9001:2008 Certified)

NITTE -574 110, Udupi District, KARNATAKA

April 2017

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum) (AICTE approved, NBA Accredited, ISO 9001:2008 Certified)

NITTE -574 110, Udupi District, KARNATAKA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled

"Voice based email system for blinds"

is a bonafide work carried out by

Diksha Kumari(4NM13CS057) *Nisha Pai (4NM13CS101)*

Parul Jha (4NM13CS110) *Pratmika Nayak (4NM13CS125)*

in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering prescribed by Visvesvaraya Technological University, Belgaum during the year 2016-2017.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide **Signature of HOD** **Signature of Principal**

Semester End Viva Voce Examination

Name of the Examiners	Signature with Date
1. _____	_____
2. _____	_____

ACKNOWLEDGEMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, our sincere thanks to our beloved principal, **Dr. Niranjan N. Chiplunkar** for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

We express our deep sense of gratitude and indebtedness to our guide, **Mr. Vijaya Murari T,** Assistant Professor, Department of Computer Science and Engineering, for his inspiring guidance, constant encouragement, support and suggestions for improvement during the course for our project. We extend our thanks to the project co-ordinators **Mr. Ranjan Kumar H S, Mr. Raju K** and **Mrs. Asmita Poojary** for their constant support.

We also thank all those who have supported us throughout the entire duration of our project.

Finally, thanks to teaching and non-teaching staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

Diksha Kumari

Nisha Pai

Parul Jha

Pratmika Nayak

ABSTRACT

In today's world, communication has become very easy due to integration of communication technologies with internet. However the visually challenged people find it very difficult to utilize this technology because of the fact that using it requires visual perception.

Even though much advancement has been implemented to help them use computers efficiently, no user who is visually challenged can use this technology as efficiently as a normal user since they require some practice for using the available technologies.

This project aims at developing an email system that will help even a naïve, visually impaired person to use the services for communication without previous training. The system does not require the use of keyboard. Instead it will work only on mouse operations and speech. This system can also be used by any normal person, for instance, by someone who is unable to read.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Internet is considered as the most important means of information and has become de facto methods used in communication. Email is one of the most common form of communication. However, it is completely useless for visually impaired and illiterate people. Currently available systems like screen readers TTS (Text-To- Speech) and ASR (Automatic Speech Recognition) does not provide full efficiency to the blind people to use internet. As nearly 285 million people worldwide are visually impaired so it is necessary to make internet facilities for communication usable for them.

Therefore, in this project 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 with ease. All the functions are based on simple mouse click operations 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.

1.1 OVERVIEW

This project is a prototype application for a standalone user which works on all the system having Python 2.7 installed.

When the user runs the application, welcome page will be displayed. The welcome page is where the user will have to provide his username and password for authentication. Upon successful login, the Home page is displayed containing several options. The voice engine will keep prompting the user as to what operation is to be performed based on hover action. The user just has to traverse across the screen and upon hearing the prompt for the action he wishes to perform; he should click the mouse button. Once the user is done with the particular module, he can return to the Home page by clicking the “Return to Home page”.

1.2 PROBLEM STATEMENT

We aim to overcome the shortcomings of the existing system in making the user completely independent with our proposed system.

1.3 STUDY AREAS

Our area of studies include python and its various packages

1.4 OBJECTIVE

This project aims at developing an email system that will help even a naïve, visually impaired person to use the services for communication without previous training. The system does not require the use of keyboard. Instead it will work only on mouse operations and speech conversion to text. This system can also be used by any normal person, for instance, by someone who is unable to read.

1.5 METHODOLOGY

The complete system is based on voice prompts and click events. 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, in most of the part user won't require to use the keyboard. All operations will be based on mouse click events. . Now the question that arises is that how will the blind users find location of the mouse pointer. As particular location cannot be tracked by the blind user, therefore the user has to traverse the mouse throughout the screen from top to bottom and then left to right. This system will be perfectly accessible to all types of users as it is just based on simple mouse clicks and there is no need to remember keyboard shortcuts. Also because of this facility those who cannot read need not worry as they can listen to the prompting done by the system and perform respective actions.

1.6 ORGANISATION OF THE REPORT

This report is divided up into chapters, each dealing with different aspects of the project. Each chapter has a short introduction, explaining the subject of each chapter, and then the details each module is explained separately. The following is a short overview of each of the chapters:

Chapter 2: Outlines some of the research made on the project in the beginning. More research was made as this project report was being developed, as new areas had to be investigated. This research is summarized in the various chapters according to the different modules.

Chapter 3: Specifies the software requirements specification, the existing and the proposed system.

Chapter 4: Specifies the Software approaches used in the project.

Chapter 5: Gives the architecture diagram, use case diagram and sequence diagram of the modules and also gives an outline of the design of the system.

Chapter 6: Specifies implementation design used in the project.

Chapter 7: The various testing such as Unit testing, System testing and Integration testing is discussed here.

Chapter 8: Shows the Results and Discussion of the project.

Chapter 9: The conclusion and the future scope of the project is discussed here.

CHAPTER 2

LITERATURE SURVEY

This chapter provides an introduction to the areas of research. It describes the work which has already been done in direct-show and states the new scope. The scope has been clearly explained and the technology used to obtain this has been mentioned in this chapter.

2.1 EXISTING SYSTEM

There are 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 common form of communication. The most common mail services that we use in our day to day life cannot be used by visually challenged people 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 screen readers available still these people face 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. A user is new to computer can therefore not use this service as they are not aware of the key locations.

The screen readers read out the content in sequential manner and therefore user can make out the contents of the screen only if they are in basic HTML format. Thus the new advanced web pages which do not follow this paradigm in order to make the website more user-friendly only create extra hassles for these people.

Our project aims to overcome these drawbacks.

2.2 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 system is said to be perfectly accessible only if it can be used efficiently by all types of people whether able or disable. 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. The complete system is based on voice prompt and clicks events. 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 for the most part, the user won't require the use of keyboard. All operations will be based on mouse click events. Now the question that arises is that how will the blind users find location of the mouse pointer. As particular location cannot be tracked by the blind user, therefore the user has to traverse the mouse throughout the screen from top to bottom and then left to right. This system will be perfectly accessible to all types of users as it is just based on simple mouse clicks and there is no need to remember keyboard shortcuts. Also because of this facility those who cannot read need not worry as they can listen to the prompting done by the system and perform respective actions.

CHAPTER 3

SYSTEM ANALYSIS & REQUIREMENTS

Analysis is the process of breaking a complex topic into smaller to get better understanding of it. Here analysis had been done based on the three aspects: System analysis, Requirement analysis and Functional requirements. System analysis comprises of relevance platform and relevance programming language. Requirement analysis reveals input, output, scope and boundary, user objective, assumptions and constraint. Functional requirements specifies about hardware and software requirements.

3.1 SYSTEM ANALYSIS

Here the analysis of the system is made with respect to relevance of platform, programming languages.

3.1.1 Relevance of platform

The application can work with the all Python enabled systems with version 2.7

3.1.2 Relevance of Programming Language

Python is a widely used high-level programming language for general-purpose programming. An interpreted language, Python has a design philosophy which emphasizes code readability and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale.

Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing python code to run on a wide variety of systems. CPython, the reference implementation of python, is open source software and has a community based development model, as do nearly all of its variant implementation.

Python is multi-paradigm programming language: object-oriented programming language and structured programming are fully supported and many language features support functional programming and aspect oriented programming. An important feature of Python is dynamic name resolution (late binding) which binds method and variable names during program execution.

3.2 REQUIREMENT ANALYSIS

Requirement analysis explains the scope and boundary of the project, user objectives, inputs and outputs of the project. It consists of assumptions and constraints.

3.2.1 Scope and Boundary

The proposed system will meet the needs of the user by providing following features:

a. The visually impaired person can efficiently access the mail services.
b. This system will focus more on the user friendliness of all types of people including normal, visually impaired as well as illiterate people.
c. The system will be highly efficient and easily understandable because it will make use of simple mouse clicks only and not keyboard shortcuts.
d. The user can read the mail, record the mail and send the mail.
e. The project can work on any system having Python 2.7 installed including Windows, LINUX and Mac OS.

3.2.2 User Objective

The user objective is to access the mail services efficiently by simple mouse clicks and perform different operations.

3.2.3 Inputs and Outputs

To login to the account, the user needs to provide the specified username and password. Once the user authentication is done, he will be directed to the home page. If he is a new user, then he needs to sign up to the mail account by providing his details and registering to it. When the user wants to compose the message to be sent, he needs to record the message using a microphone and then the mail will be sent. The user can view the mail sent by him in the ‘sent mail’ module. The user can read all the mails he has received in the ‘inbox’ module.

3.2.4 Assumptions and Dependencies

- This application is compatible with Windows and Linux OS systems with 2.7.13 version of Python along with the required packages.
- The application is to be started in order for the user to start using it.
- Some help maybe needed by the user to log in to the account or to sign up to the application.

3.2.5 General Constraints

- Required packages need to be installed in the system with the aforementioned version of Python.
- All audio files should be present in the system.
- A proper headphone is to be used with a working microphone.
- Detachable mouse is required.

3.3 FUNCTIONAL REQUIREMENTS

Functional requirements consist of two types. Those are software and hardware requirements.

3.3.1 Requirements for Application

3.3.1.1 Software Requirement

Software Requirements are:

- Python:
    1. Version: 2.7.13
    2. Requirements: Windows (All Versions), Linux
    3. Languages: Multiple languages
    4. License: Open Source
    5. Packages: Tkinter, mp3play, pyaudio,wave

3.3.1.2 Hardware Requirement

Hardware Requirements are:

- Mouse
- Headphone with microphone jack

3.4 NON FUNCTIONAL REQUIREMENTS

Performance Requirements:

Application requires working system with the specified software and hardware requirements.

Reliability:

Application can be used via any system from any location and at any time.

Availability:

Application can be made use of at any time in the system having Python and its relative packages installed.

Maintainability:

Maintenance is easy and economical.

Portability:

This system can be run on any operating system including Windows, Linux and Mac.

Security Requirements:

Application requires user to login with his mail credentials as specified in code.

CHAPTER 4

SOFTWARE APPROACH

Software approach is the sequence of steps involved in the development process along with the software used in each step.

4.1 ABOUT PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

4.2 ABOUT PyAudio

PyAudio provides Python bindings for PortAudio, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms. PyAudio is inspired by:

- pyPortAudio/fastaudio: Python bindings for PortAudio v18 API.
- tkSnack: cross-platform sound toolkit for Tcl/Tk and Python.

To use PyAudio, first instantiate PyAudio using pyaudio.PyAudio(), which sets up the portaudio system. To record or play audio, open a stream on the desired device with the desired audio parameters using pyaudio.PyAudio.open(). This sets up a pyaudio.Stream to play or record audio. Play audio by writing audio data to the stream using pyaudio.Stream.write(), or read audio data from the stream using pyaudio.Stream.read(). Note that in “blocking mode”, each pyaudio.Stream.write() or pyaudio.Stream.read() blocks until all the given/requested frames have been played/recorded. Alternatively, to generate audio data on the fly or immediately process recorded audio data, use the “callback mode” outlined below use pyaudio.Stream.stop_stream() to pause playing/recording, and pyaudio.Stream.close() to terminate the stream. Finally, terminate the portaudio session using pyaudio.PyAudio.terminate().

In callback mode, PyAudio will call a specified callback function whenever it needs new audio data (to play) and/or when there is a new (recorded) audio data available. Note that PyAudio calls the callback function in a separate thread. The function has the following signature callback (<input_data>, <frame_count>, <time_info>, <status_flag>) and must return a tuple containing frame_count frames of audio data and a flag signifying whether there are more frames to play/record.

Start processing the audio stream using pyaudio.Stream.start_stream(), which will call the callback function repeatedly until that function returns pyaudio.paComplete. To keep the stream active, the main thread must not terminate, e.g., by sleeping.

CHAPTER 5

SYSTEM DESIGN

The system design is the process of defining the architecture, components, modules, interfaces and data for a system to satisfy requirements. Systems design could be seen as the application of systems theory to product development. The design is broadly classified into two levels according to software engineering. They are high level design and low level design. They are explained as follows.

5.1 GENERAL DESIGN ARCHITECTURE

The general design architecture of the system depicts the interaction of the subsystems in the system as shown in figure 5.1

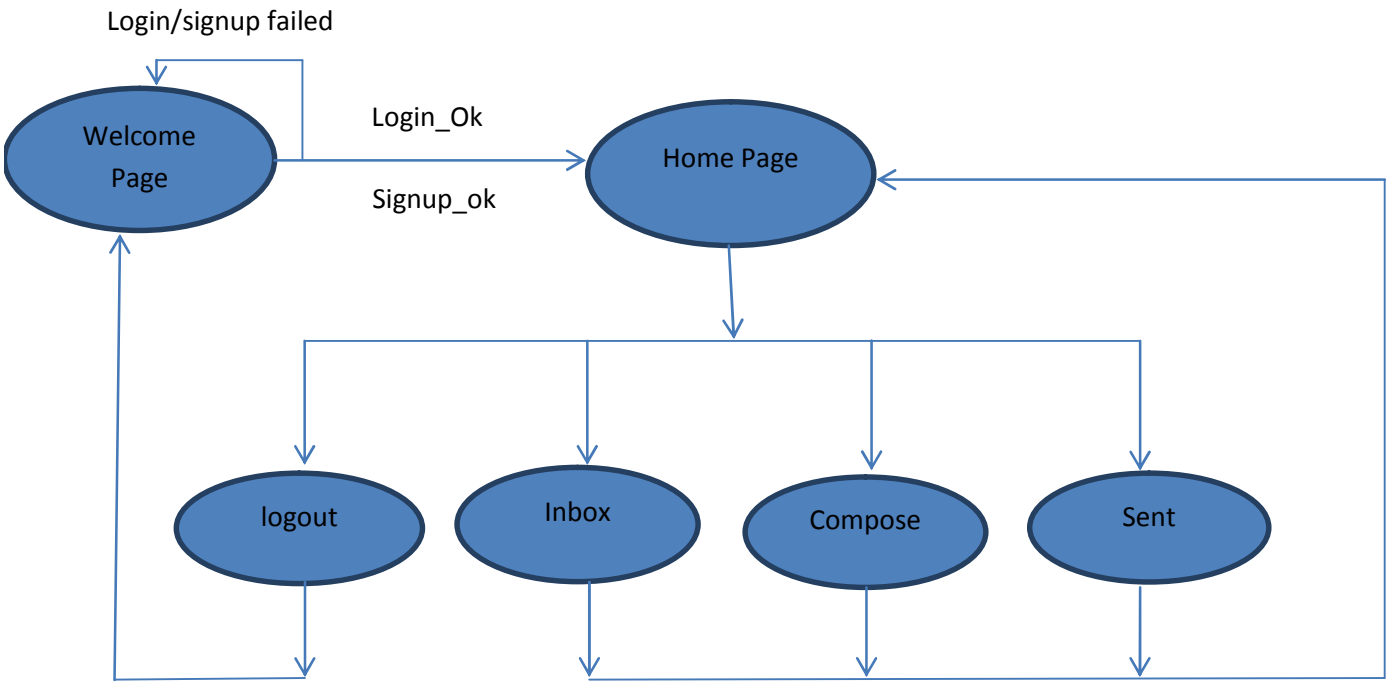

Figure 5.1: General design architecture

5.1.1 Sequence Diagram

A sequence diagram is a kind of interaction diagram that shows how process operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence as shown in figures below.

5.1.1.1 Scenario 1

Once the application is up and running, the user has two options – to sign up (for new user) or to log in (for registered users). If the user wishes to sign up, they will be directed to the Registration Page where they have to enter their name, password and re-enter the password. On clicking the Registe button, the two passwords are checked and if there is a mismatch, an error message is displayed. On the other hand, if there is a match, the user's Home Page will be displayed. If the user wishes to log in to their account, they have to enter the name and the password and on proper authentication, they are directed to the respective Home Page. In case of mismatch, again, error message will be displayed. On clicking the Logout button, the user is signed out of the account and the Welcome Page is displayed.

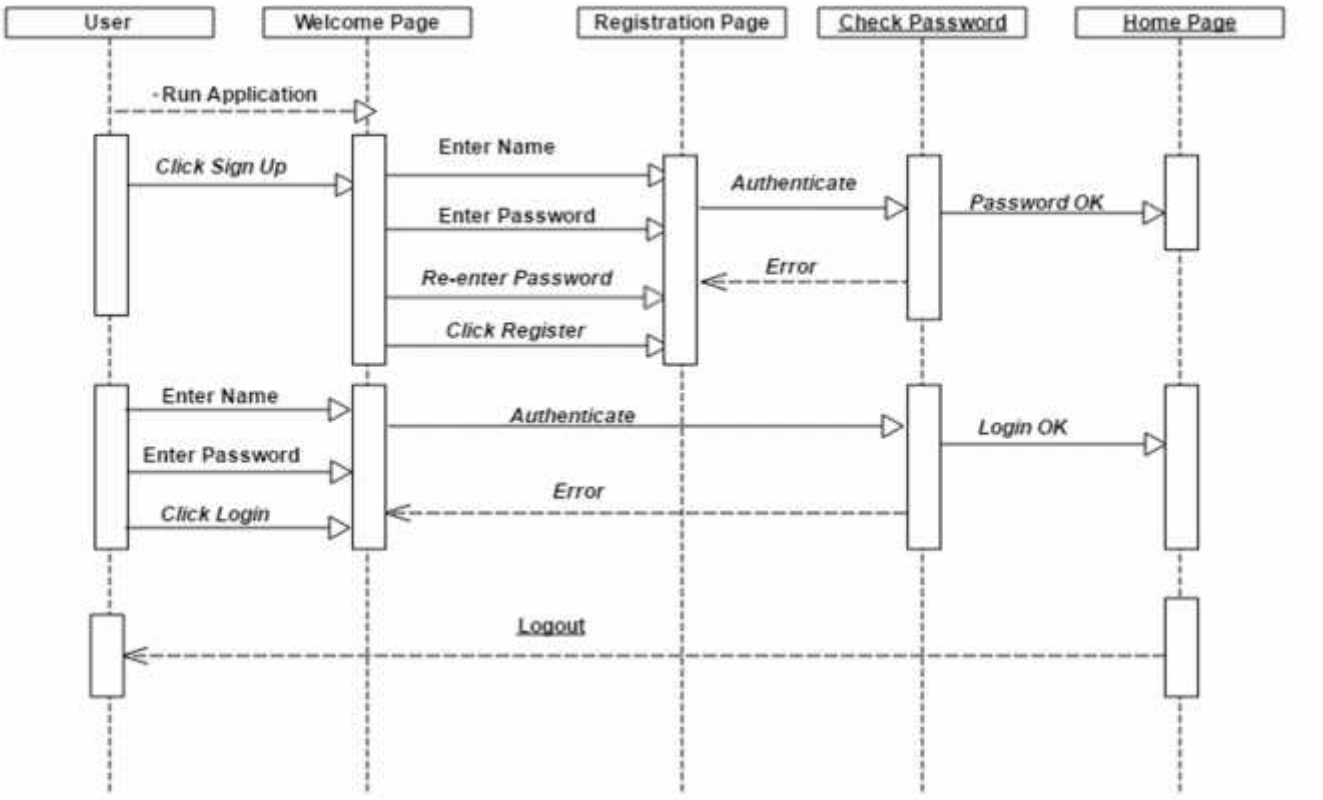

Figure 5.1.1.1: Sequence diagram for Scenario1

5.1.1.2 Scenario 2

Once the user is on the Home Page, they have 4 options namely, Inbox, Compose, Sent mail, Log out. If the user wishes to check the received mails, the Inbox button is to be clicked. This page has several buttons, each corresponding to one received mail. Based on the mail they wish to read, click on the respective button. Once the user is done with the inbox, Return to HomePage button is to be clicked to go back to the Home Page. If a mail is to be composed, Compose option is to be clicked. Within this module, the user has 3 options – record a message, listen to the recorded message, send the message. On clicking on Record, the voice input from the user is recorded for a definite period of time. This recorded message can be played by the user to check if he is satisfied with it. For this, the Listen button is to be clicked upon. On hitting Send, the mail is then sent and the user is directed back to the Home Page. Another option available to the user in the Home Page is the Sent mail viewing option. Here, the user can replay the message that had just been recorded and sent. Once this is done, the user can return to the Home Page by clicking the button for the same. On clicking Logout, the user is signed out of his account and redirected to the Welcome Page of the application.

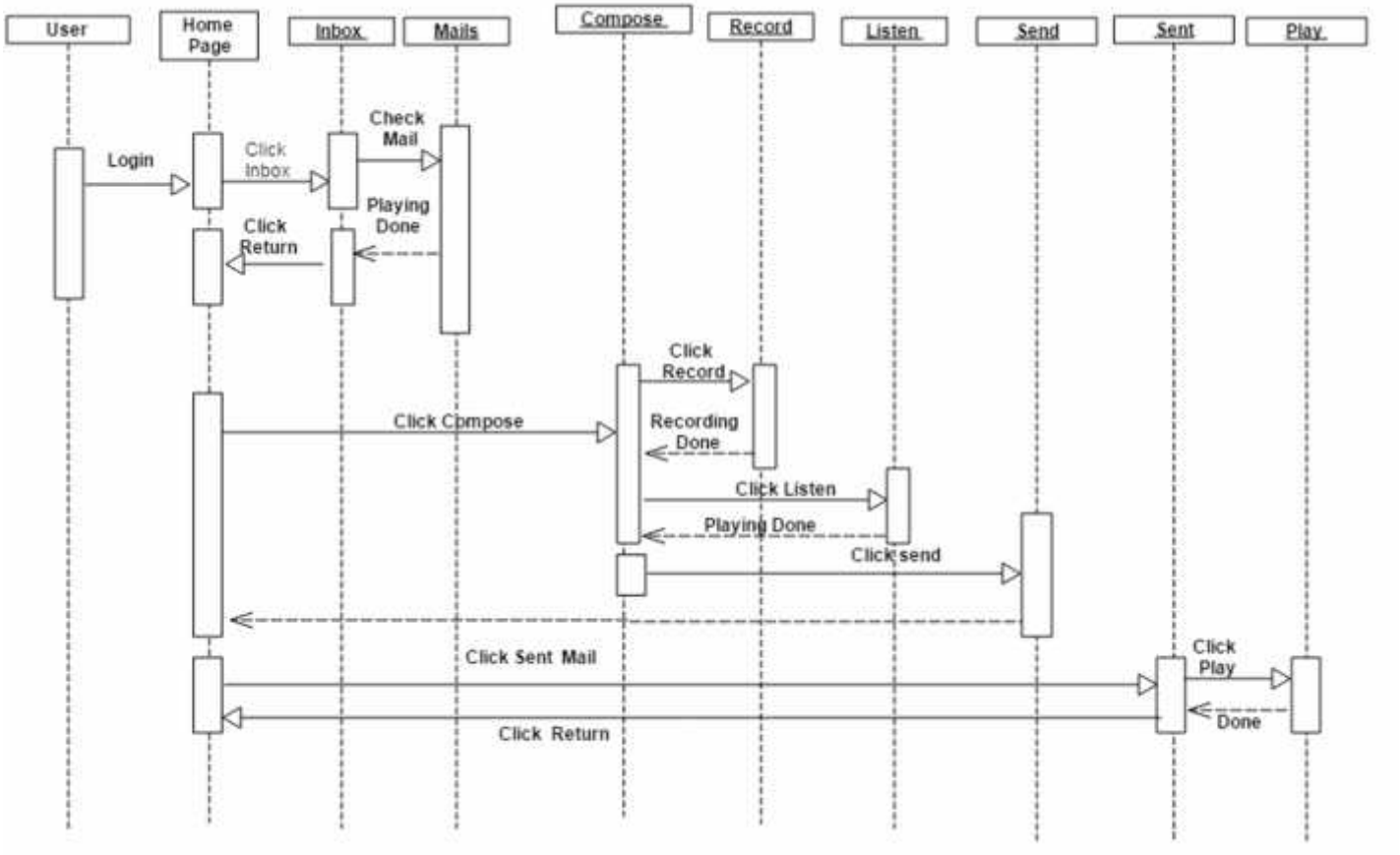

Figure 5.1.1.2: Sequence diagram for Scenario2

5.1.2 Activity Diagrams

Activity diagrams are graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Fig. 5.1.2 we show the activity diagram for different modules.

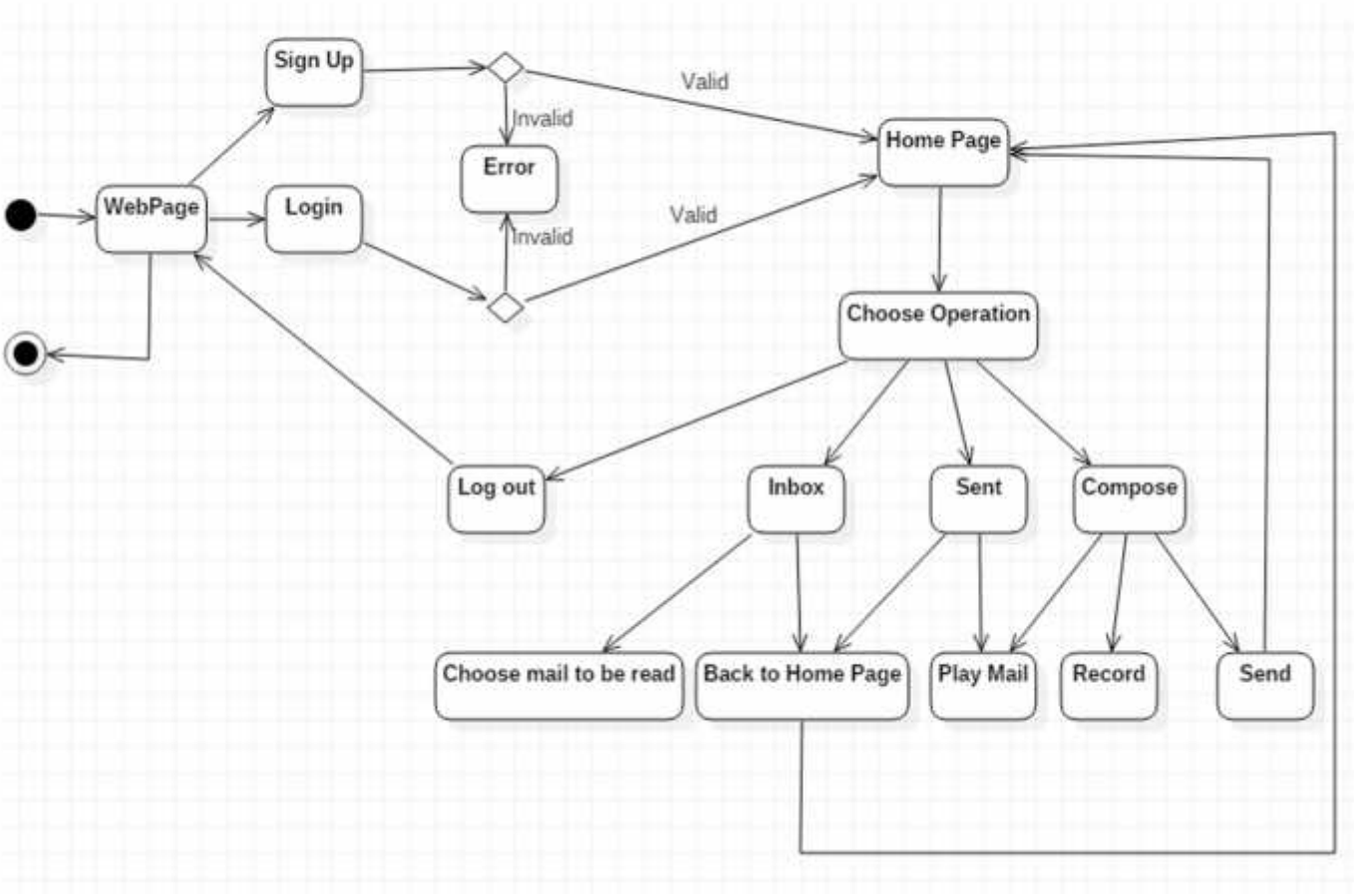

Figure 5.1.2: Activity Diagram

5.1.3 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and various ways that they interact with the system as shown in Fig. 5.1.3.

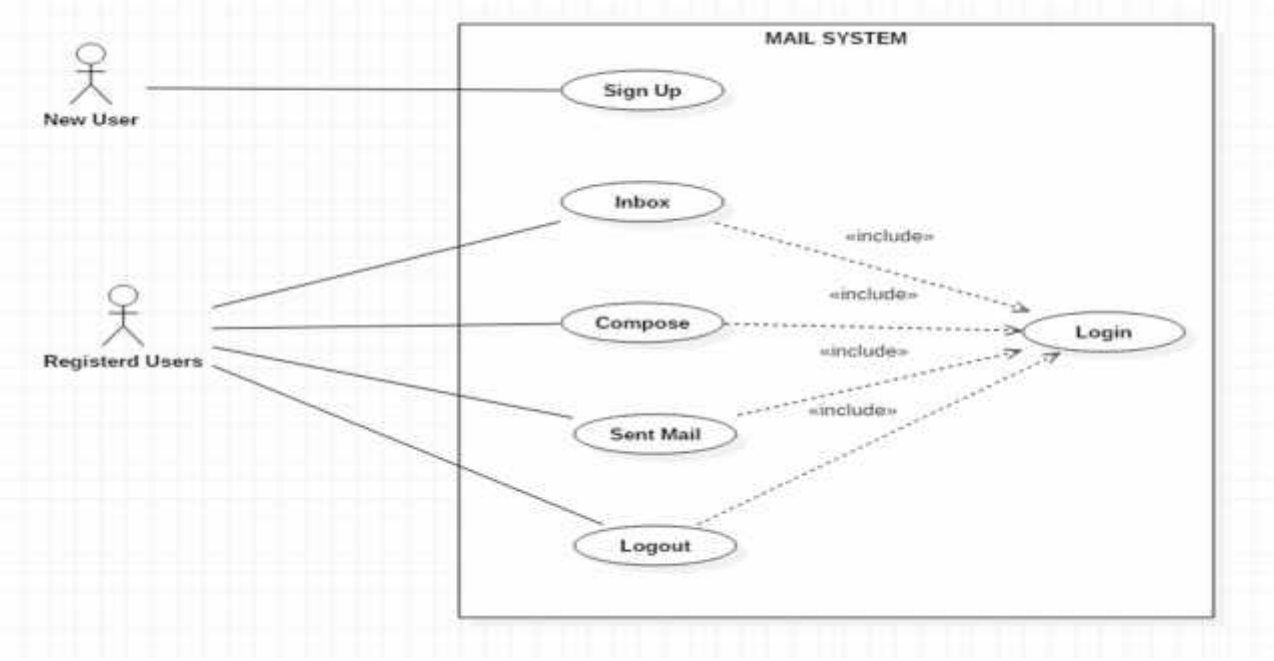

Figure 5.1.3.1: Use case for Mail system

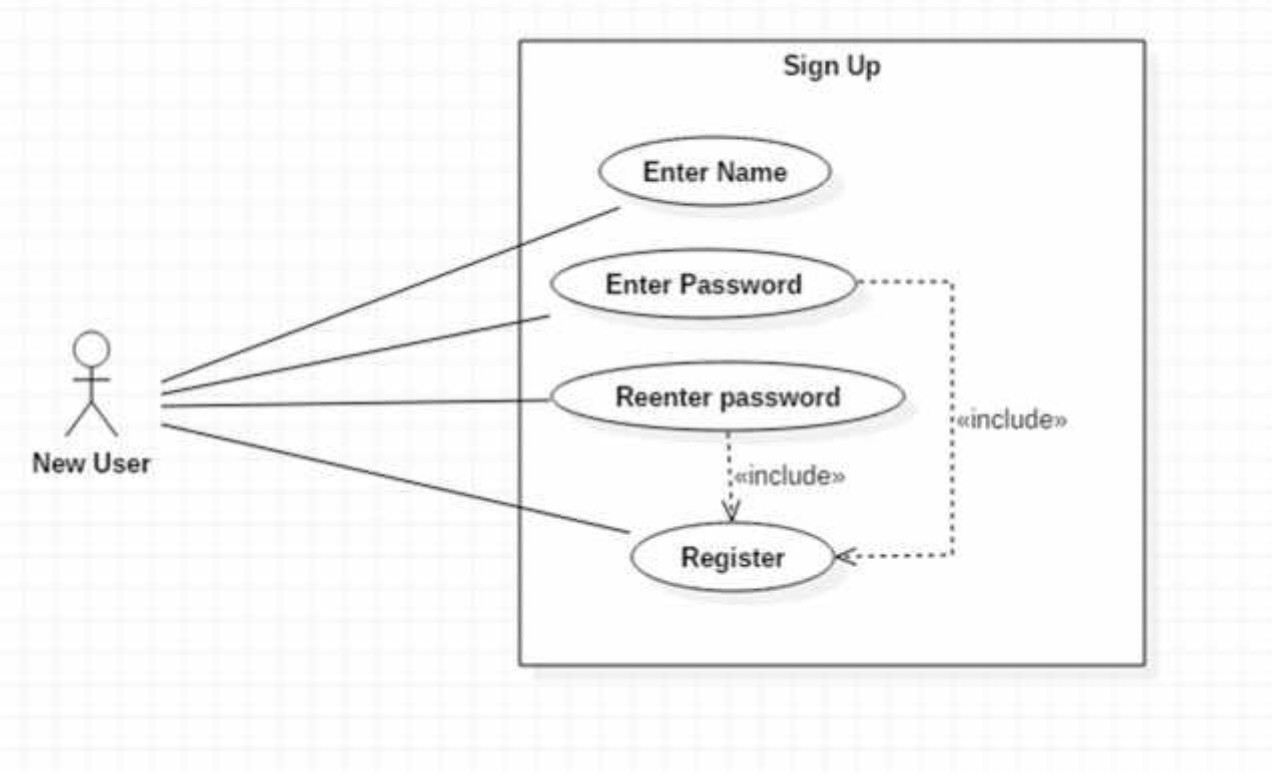

Figure 5.1.3.2: Use case for Sign Up

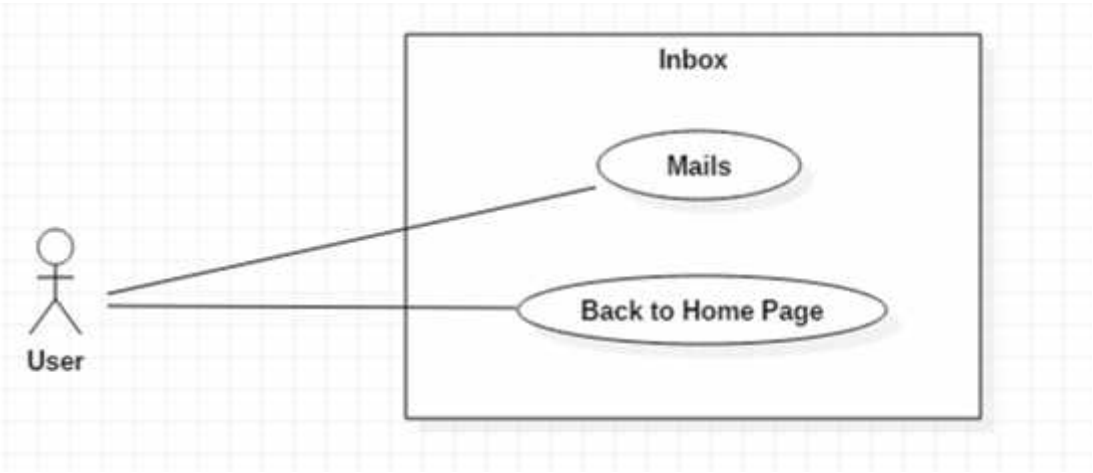

Figure 5.1.3.3: Use case for Inbox

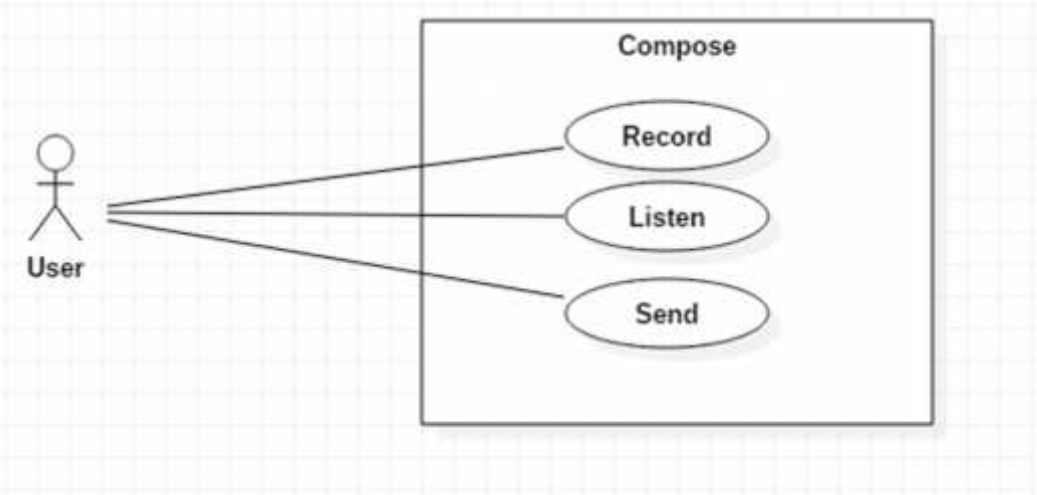

Figure 5.1.3.4 Use case for Compose

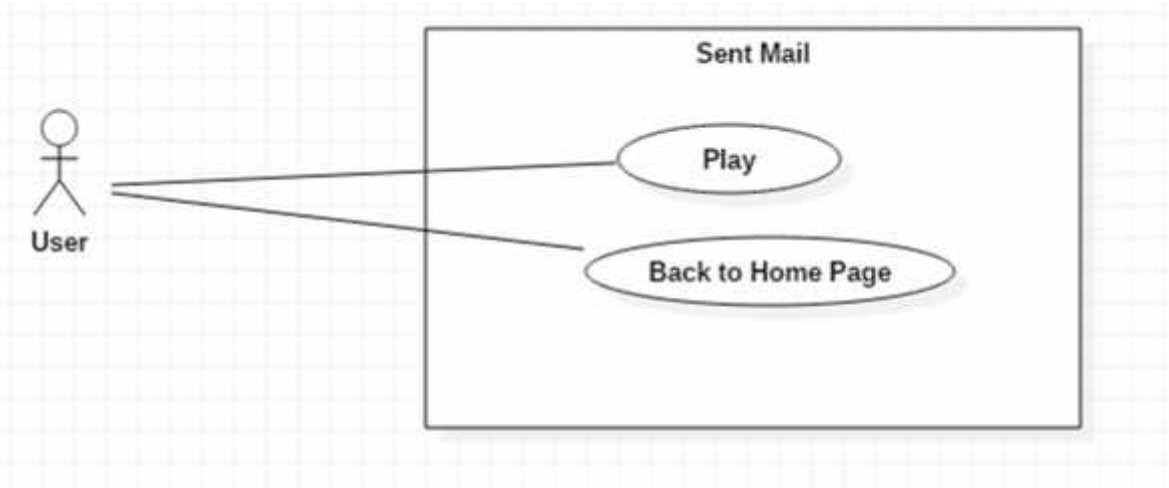

Figure 5.1.3.5 Use case for Sent mail

CHAPTER 6

SYSTEM IMPLEMENTATION

The basic goal in system implementation is to specify the logic for the different modules that have been specified during system design. Specifying the logic will require developing an algorithm that will implement the given specification. This chapter gives the implementation details of the entire system.

6.1 SOFTWARE APROACH

6.1.1 About PyAudio

PyAudio provides Python bindings for PortAudio, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms. PyAudio is inspired by:

- pyPortAudio/fastaudio: Python bindings for PortAudio v18 API.
- tkSnack: cross-platform sound toolkit for Tcl/Tk and Python.

To use PyAudio, first instantiate PyAudio using pyaudio.PyAudio(), which sets up the portaudio system. To record or play audio, open a stream on the desired device with the desired audio parameters using pyaudio.PyAudio.open(). This sets up a pyaudio.Stream to play or record audio. Play audio by writing audio data to the stream using pyaudio.Stream.write(), or read audio data from the stream using pyaudio.Stream.read(). Note that in “blocking mode”, each pyaudio.Stream.write() or pyaudio.Stream.read() blocks until all the given/requested frames have been played/recorded. Alternatively, to generate audio data on the fly or immediately process recorded audio data, use the “callback mode” outlined below use pyaudio.Stream.stop_stream() to pause playing/recording, and pyaudio.Stream.close() to terminate the stream. Finally, terminate the portaudio session using pyaudio.PyAudio.terminate().

In callback mode, PyAudio will call a specified callback function whenever it needs new audio data (to play) and/or when there is a new (recorded) audio data available. Note that PyAudio calls the callback function in a separate thread. The function has the following signature callback (<input_data>, <frame_count>, <time_info>, <status_flag>) and must return a tuple containing frame_count frames of audio data and a flag signifying whether there are more frames to play/record.

Start processing the audio stream using pyaudio.Stream.start_stream(), which will call the callback function repeatedly until that function returns pyaudio.paComplete. To keep the stream active, the main thread must not terminate, e.g., by sleeping.

6.1.2 About WAV

Short for WAVeform Audio Format, it is normally used in an uncompressed format on the Microsoft Windowsplatform. This raw audio format, which was developed jointly by IBM and Microsoft, stores audio data in blocks. On the digital music scene, its usefulness has diminished over time with the development of better lossless audio formats, such as FLAC and Apple lossless. It is a standard that will probably be used for some time yet due to its widespread use in professional music recording and is still a very popular format for audio/video applications.

The file extension associated with WAV is: .WAV

6.1.2.1 About WAV in Python

The wave module provides a convenient interface to the WAV sound format. It does not support compression/decompression, but it does support mono/stereo.

The wave module defines the following function and exception:

wave.open(*file*[, *mode*])

If file is a string, open the file by that name, otherwise treat it as a seekable file-like object. Mode can be of:
'r', 'rb' - Read only mode.
'w', 'wb' - Write only mode.
Note that it does not allow read/write WAV files.

A mode of 'r' or 'rb' returns a Wave_read object, while a mode of 'w' or 'wb' returns a Wave_write object. If mode is omitted and a little file-like object is passed as a file, file mode is used as the default value for the mode (the 'b' flag is still added if necessary).

If you pass in a file-like object, the wave object will not close it when its close() method is called; it is the caller's responsibility to close the file object.

wave.openfp(*file*, *mode*)

A synonym for open(), maintained for backwards compatibility.

exception wave.Error

An error raised when something is impossible because it violates the WAV specification or hits an implementation deficiency.

6.1.3 About mp3play

A simple interface for playing music from an MP3 file. Allows your Windows Python program to play and stop MP3s, without opening an external player or requiring any external programs. A very simple interface for the common case (playing an entire MP3), with an API for more complex tasks (e.g., playing from seconds 30 to 45 of an MP3).

6.2 MODULES

Module 1: Login and registration

The existing user can login to the account by providing the specified username and password. If the user is new to the application, then he needs to sign up to the mail account by providing his details and registering to it.

Module 2: Inbox

In the inbox module, each button represents a mail received by the user. When the user hovers over the button, he will be informed who the sender is. If he wishes to listen to the mail, he has to click the button. On clicking the respective button for the mail, the corresponding mail will be read out to him. Only after he has finished listening to the current mail, he can either listen to other mail or return back to Home Page.

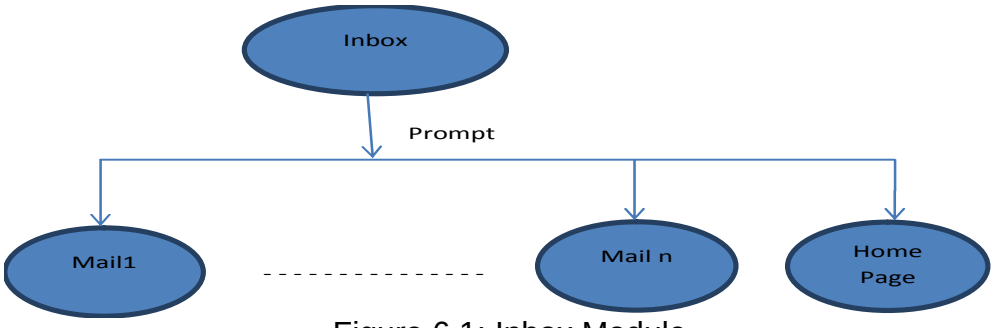

Figure 6.1: Inbox Module

Module 3: Compose

In this module, the user has three options: to record, listen and send the mail. On clicking the record button, he can record the message for a maximum time limit of 5 seconds. This recording will be saved as an audio file. He can now listen to this message by clicking on the “Listen” button. Finally, by clicking the “Send” button, the audio message will be sent to the specified recipient.

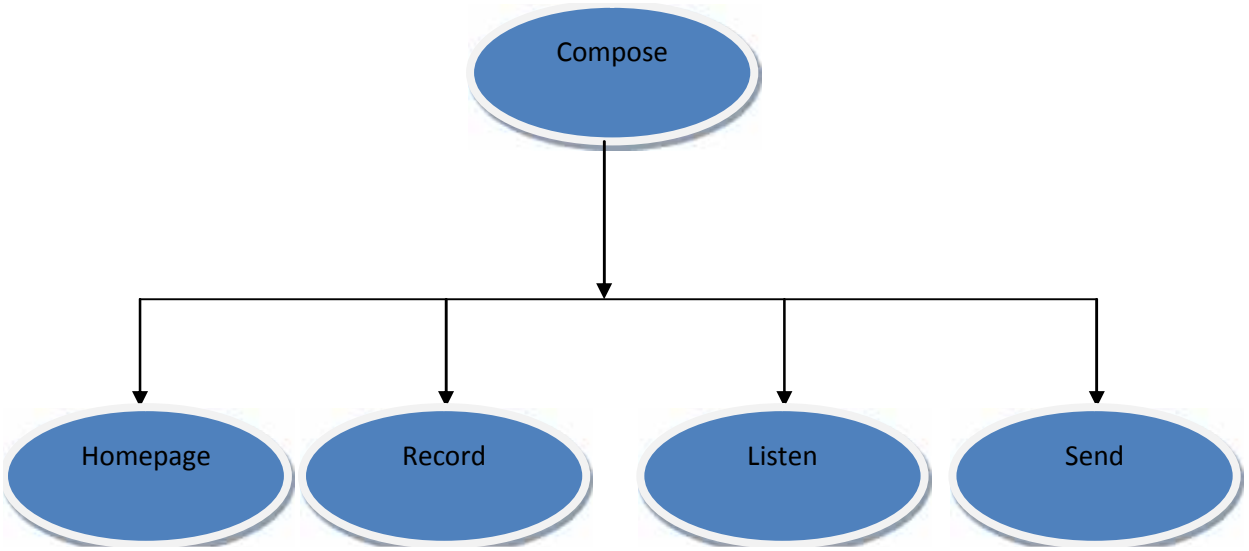

Figure 6.2: Compose Module

Module 4: Sent mail

The mail sent by the user will be shown in this module. When the user hovers over the button, he will be told to click on the button to listen to his sent mail. When he clicks the button, the most recent sent mail will be read out to him.

CHAPTER 7

SYSTEM TESTING

Testing is an important stage in the System development life cycle. Careful planning is needed to get the most out of testing and to control testing cost. Test planning is concerned with setting out standard for the testing process rather than describing the product test. System testing is an important element of software quantity assurance and ultimate review of specifications.

7.1 INTRODUCTION

Software testing is a process used to identify the correctness, completeness and quality of the developed computer software. Testing a process is questioning a product in order to evaluate it, where the questions are things the tester tries to do with the product, and the product answers with its behaviour in reaction to probing of the tester.

The testing phase is performed after the coding to detect all the errors and provide quality assurance and ensure reliability of the software. Testing is vital to the success of the system. During testing, the software to be tested is evaluated to determine if the system is performing as expected. Clearly, the success of testing in revealing errors depends critically on the test cases. The different testing strategies employed in this project are explained in this chapter.

7.2 UNIT TESTING

In computer programming, unit testing is software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is smallest testable part of an application. It is also called as module testing. The goal of unit testing is to isolate each part of the program first and then testing the sum of its parts, integration testing becomes much easier. In our project, we apply this by testing the various modules of the application and also each feature individually.

7.3 INTEGRATION TESTING

Integration testing is three phase in software testing in which individual software modules are combined and tested as group. It occurs after unit testing and before system testing. Integration testing takes as its input module that have been tested groups them in larger aggregates, applies tests defined in an integration test plan to those aggregate and delivers as its output the integrated system ready for the system testing. The purpose of the integration testing is to verify functional,

performance and reliability requirement placed on major design items. All the different modules of the project are combined and tested.

7.4 TEST CASES

Test Case	Description	Action	Expected Result	Actual Result	Result
Sign Up	Click on Sign Up button	Display the registration page	Registration Page	Same as expected	Pass
Register	Click on Register	Check if password entries match	Match: Home Page Mismatch: Error message	Same as expected	Pass
Login	Click on Login button	Check if username and password are valid	Valid: Home Page Invalid: Error message	Same as expected	Pass
Log Out	Click on Logout button	Display the Welcome Page	Welcome Page	Same as expected	Pass
Inbox	Click on Inbox button	Show the list of all the received mails	Mail<num>: Read out the received mail Return: Display Home Page	Same as expected	Pass
Compose	Click on Compose button	Show different compose options	Record: Record message for a specific period of time Listen: Play the recorded message Send: Send the message	Same as expected	Pass
Sent Mail	Click on Sent Mail button	Show the mails sent by the user	Play: Play the sent message Return: Display Home Page	Same as expected	Pass

CHAPTER 8

RESULTS & DISCUSSION

8.1 USER INTERFACE

Welcome Page:

The user can login to his account in this page by providing the correct details. Once the user provides his specified username and password and then clicks on the log in button, he is directed to the home page. If new user wants to register to the account, he has to click on sign up button and then he is directed to registration page.

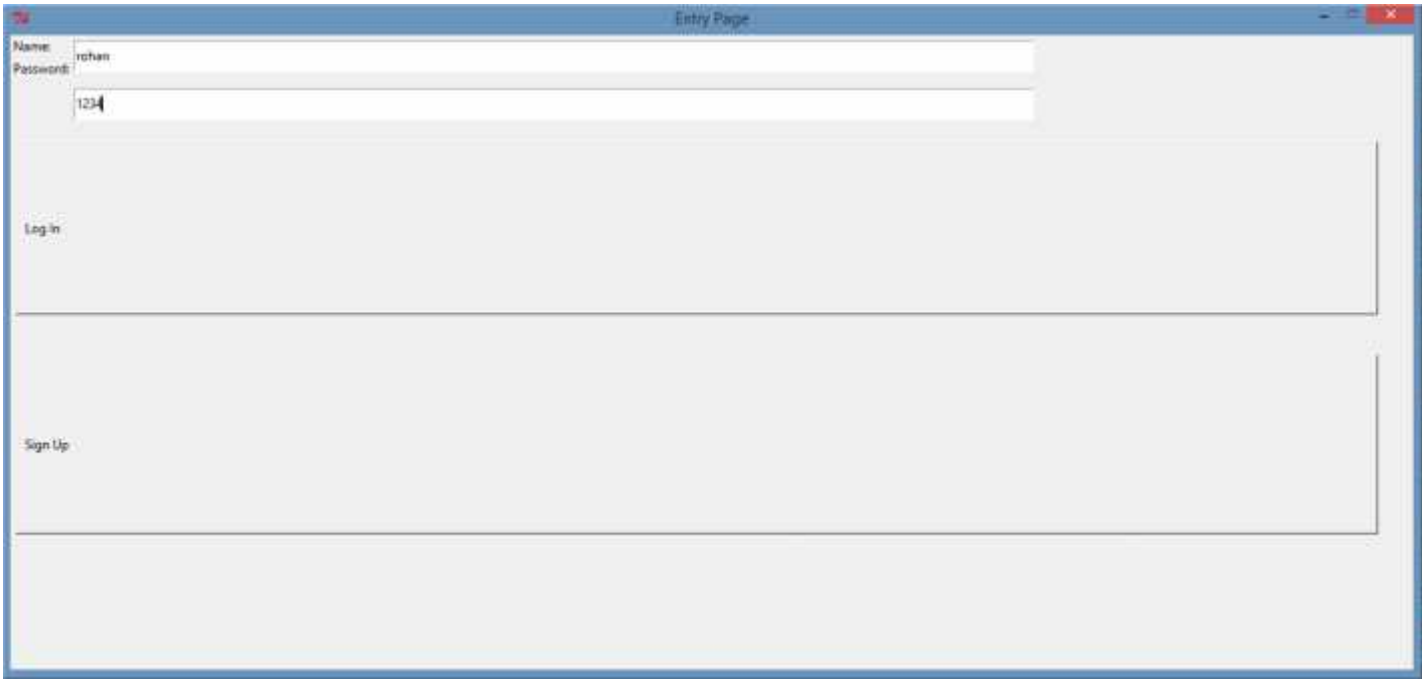

Figure 8.1.1: Wrong Login details

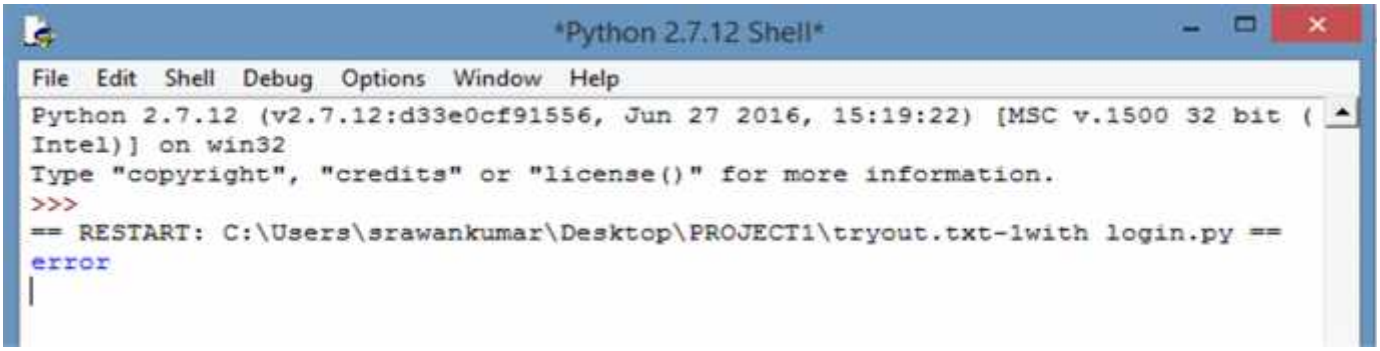

Figure 8.1.2: Error message

Figure 8.1.3: Login successful

Registration Page:

The new user who wishes to register to the account needs to provide their correct details and then click on register button. Once the authentication is done, the user can use the mail services.

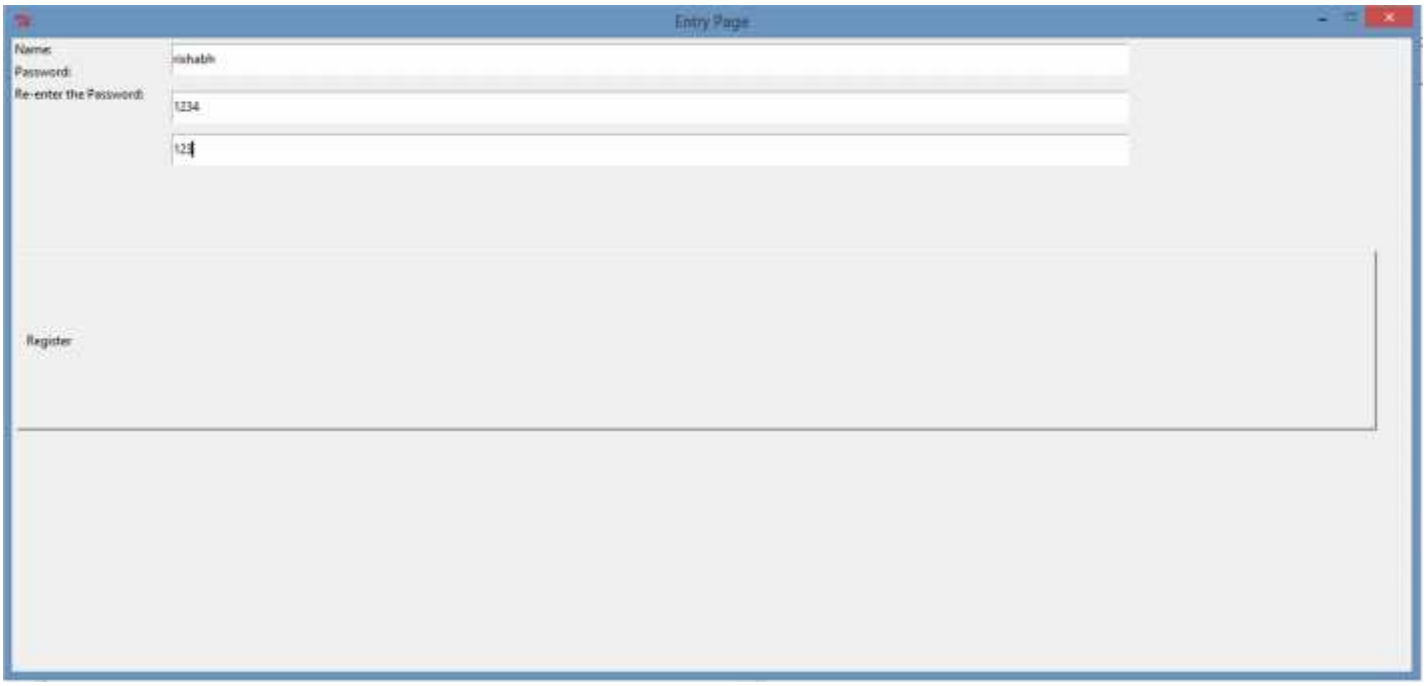

Figure 8.1.4: Wrong registration details

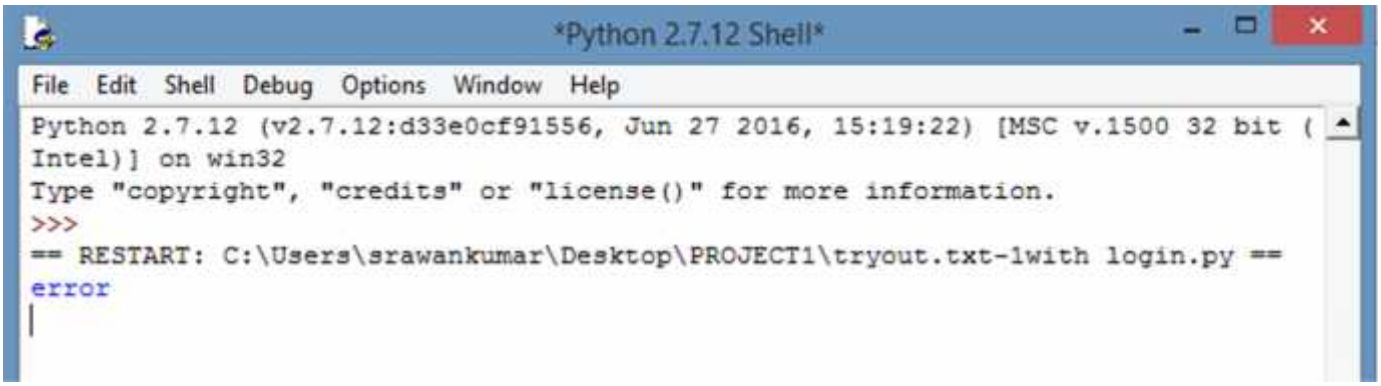

Figure 8.1.5: Error message

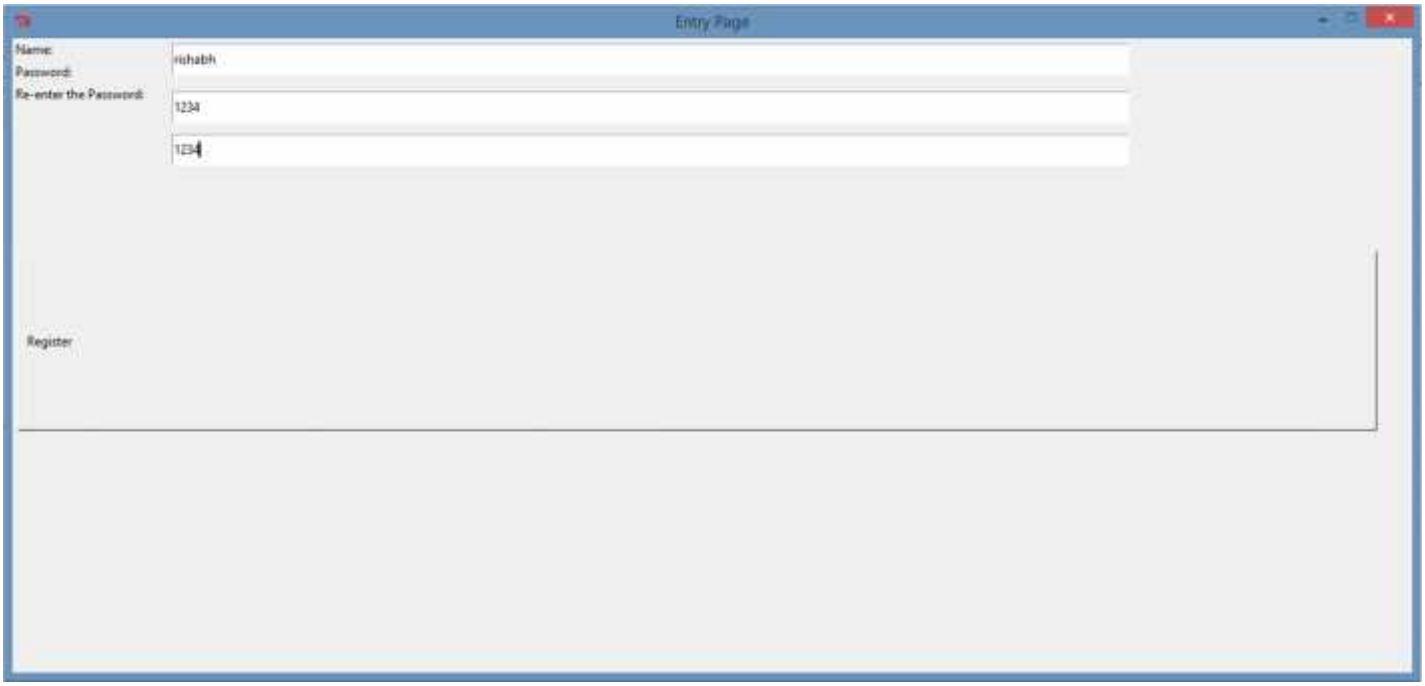

Figure 8.1.6: Registration successful

Home Page:

Once the user logs into his account, he will be directed to Home Page. Home Page consists of buttons for several different mail services like Inbox, Compose, Sent and Logout. If he wants to listen to the mails he has received, he needs to click on “Inbox” button, or if he wants to compose his message and send, he needs to click on “Compose” button, or if he wants to listen to the mail he has recently sent, he needs to click on “Sent” button. If the user wishes to logout of his account, then he can do the same by clicking on “Logout” button.

Figure 8.1.7: Home Page

Inbox:

Here, each button represents a mail received by the user. When the user hovers over the button, he will be informed who the sender is. If he wishes to listen to the mail, he has to click the button. On clicking the respective button for the mail, the corresponding mail will be read out to him. Only after he has finished listening to the current mail, he can either listen to other mail or return back to Home Page.

Figure 8.1.8: Inbox

Compose:

The user has three options: to record, listen and send the mail. On clicking the record button, he can record the message for a maximum time limit of 5 seconds. This recording will be saved as an audio file. He can now listen to this message by clicking on the “Listen” button. Finally, by clicking the “Send” button, the audio message will be sent to the specified recipient.

Figure 8.1.9: Compose

Sent Mail:

The mail sent by the user will be shown in this module. When the user hovers over the button, he will be told to click on the button to listen to his sent mail. When he clicks the button, the most recent sent mail will be read out to him.

Figure 8.1.10: Sent Mail

Logout:

When the user clicks the “Logout” button, he will be logged out of his mail account and will be directed back to the Welcome Page.

Figure 8.1.11: Logout

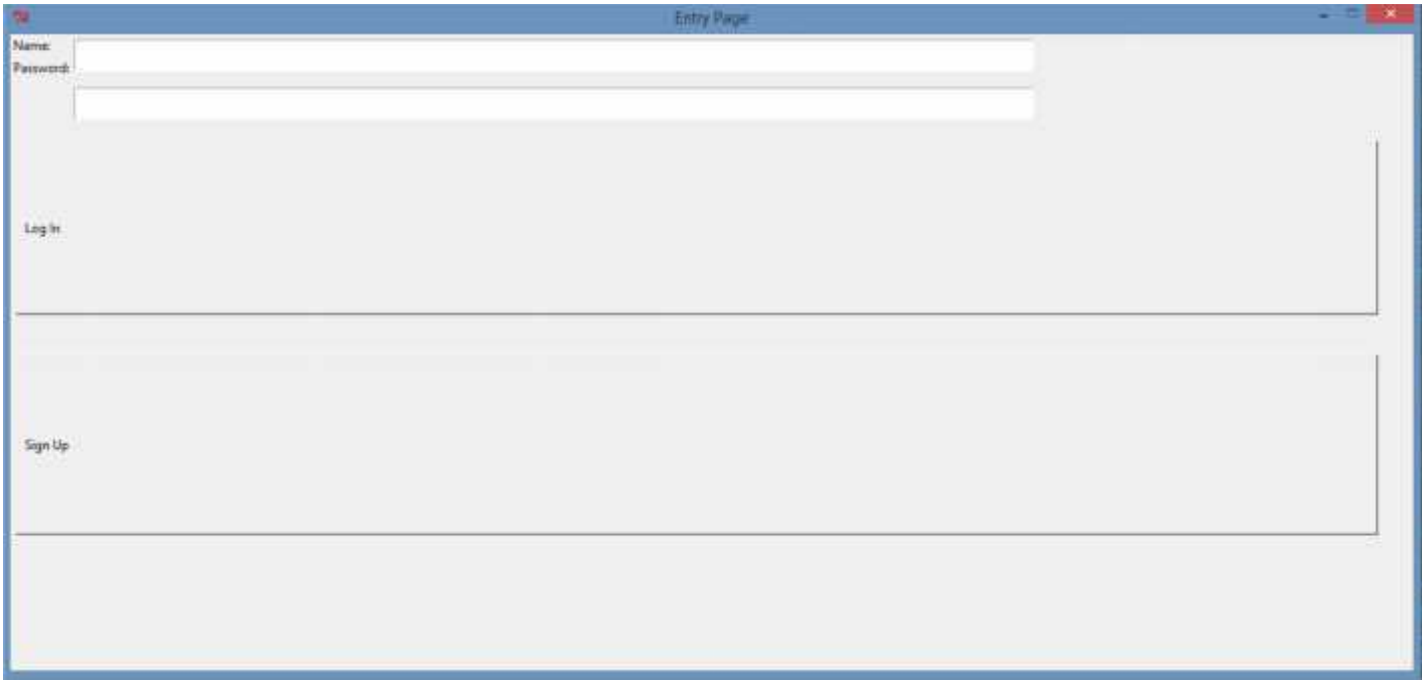

Figure 8.1.12: Welcome Page after logout

8.2 DISCUSSION

However the application fails to work as expected in the following cases:

1. If the system doesn't support the specified version of Python used for this application.
2. If the system's speaker or earphones are not working properly.
3. If wrong credentials are provided.

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 CONCLUSION

The application is designed in such a way that it succeeds in implementing the basic features of mail service, based almost entirely on mouse clicks.

Unlike current system, this system focuses more on user friendliness of all types of people including normal people visually impaired people as well as illiterate people.

9.2 FUTURE WORK

Since this application is just a prototype, there is no integration with the web. We hope to achieve the same in future for real-time purposes.

REFERENCES

Voice based email system for blinds, International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 1, January 2015.

Voice Based Email System for Blinds, International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE) Volume 1, Issue 10, March 2015. Impact Factor: 1.036, Science Central Value: 26.54.

https://www.python.org/downloads/release/python-2713/

https://pypi.python.org/pypi/mp3play

https://people.csail.mit.edu/hubert/pyaudio/docs/

https://docs.python.org/2/library/wave.html

https://www.python.org/doc/essays/blurb/