# ACKNOWLEDGEMENT

We, the students of CSE department of Geethanjali College of engineering and technology, would like to convey heartfelt thanks to **Dr D.S.R Murthy**, head of Department of computer science and engineering for the wonderful guidance and encouragement given to us to move ahead in the execution of the project.

I am extremely privileged to be to have an opportunity to work on the project under the guidance of **Mr. V Shiva Narayana Reddy,** Associate professor. We express our solicit gratitude for the valuable support which helped us a lot and successfully completing a mini project.

Above all we are very much thankful to the **management of Geethanjali College of Engineering and Technology** which was established by highly profile intellectuals for the cause of Technical Education in modern Era. We wish that GCET sooner should become a deemed university and produce unaccountable young engineers and present them in the modern technical world.

With regards,

J Ashish Reddy - 16R11A05A8

Ch Prathyusha - 16R11A05A4

S Sushant - 16R11A05D7

# TABLE OF CONTENTS

# ABSTRACT

The visually challenged people find it very difficult to access the technology because of the fact that using them requires visual perception. Even though many new advancements have been implemented to help them use computers efficiently, no user who is visually challenged can use this technology as efficiently as a normal user can do that. Unlike normal users they require some practice and also remembrance of certain keys for using the present technologies. Our approach aims at developing an email system that will help even a visually impaired person to use the services for communication without any prior training. This system will also reduce the load taken by blind to memorize and type characters using keyboard. This system will be keyboard independent and will work only Speech recognition. The system is completely based on interactive voice response (IVR) which will make it user friendly and efficient to use.

# List of Figures

# List of Screens

# CHAPTER – 1
# INTRODUCTION

## 1.1 Area of Project

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.

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.

Interactive voice response (IVR) is a technology that allows a computer to interact with humans through the use of voice. Speech recognition is the inter-disciplinary sub-field of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text by computers. It is also known as “automatic speech recognition” (ASR), “computer speech recognition”, or just “speech to text” (STT). Speech recognition works using algorithms through acoustic and language modelling. Acoustic modelling represents the relationship between linguistic units of speech and audio signals; language modelling matches sounds with word sequences to help distinguish between words that sound similar.

Our Email system using speech recognition could potentially serve as an efficient input method for mailing devices for blind. The application will be an Interface for visually

impaired persons using IVR- Interactive voice response. Thus, enabling everyone to control their mail accounts using their voice only and to 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. The main benefit of this system is that the use of keyboard is eliminated, the user will have to respond through voice and mouse click only.

## 1.2 Objectives

This project proposes a python-based application, designed specifically for visually impaired people. This application provides a voice-based mailing service where they could read and send mail on their own. This mailing system can be used by a blind person to access mails easily and adeptly.

Hence dependence of visually challenged on other individual for their activities associated to mail can be condensed. The application will be a python-based application for visually challenged persons using IVR- Interactive voice response, thus sanctioning everyone to control their mail accounts using their voice only and to be able to read, send.

The main advantage of this system is that use of keyboard is eliminated, the user will have to respond through voice only.

# CHAPTER – 2

# SYSTEM ANALYSIS

## 2.1 Existing System

The most common mail services that we use in our day to day life cannot be used by visually challenged people. This is because they do not provide any facility so that the person in front can hear out the content of the screen. As they cannot visualize what is already present on screen they cannot make out where to click in order to perform the required operations. For a visually challenged person using a computer for the first time is not that convenient as it is for a normal user even though it is user friendly.

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

Another drawback that sets in is that 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. All these are some drawbacks of the current system which we will overcome in the system we are developing.

## 2.2 Proposed Systems

### 2.2.1 Details

The planned system is relying on a very fresh plan and obscurity just like the accessible mail systems. The present systems don't give this much convenience. So, the systems present have a tend to area unit developing is totally dissent from this system. In contrast to present system which emphasize more on user easiness of naive users, this system focuses more on user easiness of all kind of folks including naive folks visually disabled people as well as uneducated people. The entire structure is based on IVR- interactive voice response. When using this system, the computer will prompt the client to perform precise operations to gain relevant services and if the client needs to way in the relevant services then they need to perform that operation.

One of the most important recompense of this system is that user will not need to use the keyboard. All operations will be based on voice proceedings.

### 2.2.2 Impact on the Environment

With the existing methods, it was difficult for the visually impaired people to use the Email systems through which they were fallen behind in this fast and mass communication process. Using this proposed system, they can cope with others to send and receive emails to communicate with others using their audio as input.

### 2.2.3 Safety

Safety is a critical part in the emails as a lot of information is transferred now a days through emails. In this Email system, the entered credentials that is the username and password are sent to the Gmail server and a connection with server is established only if the entered username and password are correct. If not, connection establishment fails. In this process there is no loss of data or any malpractices like hacking is also not possible as the information in the email can be accessed only if the connection is established.

### 2.2.4 Ethics

In this Email system, if the connection with server is not established then one cannot access any personal information. Irrespective of the establishment the connection with server there is no kind of physical or virtual harm is not done to anyone.

### 2.2.5 Cost

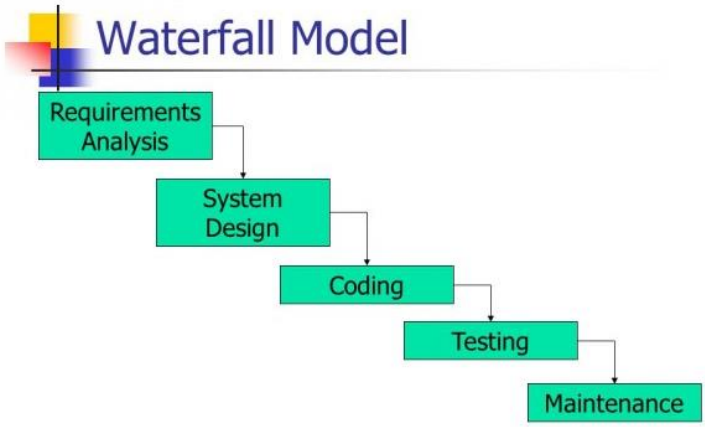
In this Proposed system, it's a low-cost Project where services provided, and methodologies involved in this system are of low cost. There won't be any maintenance cost also.

### 2.2.6 Type

The proposed system is a type of Console interface. The execution of this project is done on any python console. Any console having all the required library files installed can execute this Voice Based Email System without any problem.

### 2.2.7 Standards

The software model used for system is waterfall model. waterfall model is a systematic and sequential approach to the software development. This includes system engineering and modelling which establishes requirements for all system elements and allocating some subset of this requirement to software. System engineering and analysis is requirement gathering at the system level with small amount of top-level design.



**Problem Definition**

This is the very first stage to developing the project. It defines the Aim and the concept of the project. The aim of "Voice-Based Email System for Blind People" is providing effective accessing capabilities to people having reading disabilities visual impairments.

**Requirement Analysis**

Existing systems such as screen readers and ASR are analyzed. Care is taken to ensure that all the drawbacks of the existing systems were overcome.

**Design and Coding**

It is necessary to get logical flow of the software. Program must be efficient enough to execute faster.

**Testing**

Testing will involve working of individual model after integration and the working of whole project.

**Maintenance**

Maintenance of system is to check if all speech commands are working as expected all the time. A regular check must be made on these.

## 2.3 Scope of the Project

For people who can see, e-mailing is not a big deal, but for people who are not blessed with gift of vision it postures a key concern because of its intersection with many vocational responsibilities. This voice-based email system has great application as it is used by blind people as they can understand where they are. E.g. whenever cursor moves to any icon on the website say Register it will sound like "Register Button". There are many screen readers available. But people had to remember mouse clicks. Rather, this project will reduce this problem as

mouse pointer would read out where he/she lies. This system focuses more on user friendliness of all types of persons including regular persons, visually compromised people as well as illiterate. Voice could be extended to image attachments and other options such as indentation, fonts etc., that are available with normal E-Mail.

## 2.4 Modules Description

**A**. Phase-1:

The tasks that can be performed using the program developed will be prompted using the voice prompt. In background python module pyttsx3 is used for text to speech conversion. User will be asked to provide input for the following tasks written below. The input is expected in the form of speech by the user which will be converted to text by the Google speech application interface in python and accordingly tasks will be performed.

• Login to their Gmail account.

• Send e-mail through Gmail.

• Read e-mail through Gmail.

**B**. Phase-2:

In phase-2 of our program the user will give speech input to the system. This speech input will be handled by speech recognition module. It is a python library which is used to handle the voice requests and it converts speech into text. Now after receiving input from the user speech to text converter will save the response in respective variables used in the script and based on their value it will further enter respective modules.

**C**. Phase-3:

In this phase our program will handle the requests by the user. Based on the speech input given by the user it will launch the modules.

• Login to G-mail account: - This module will handle the request by user to login in their g-mail account. This module will make the connection with the user's Gmail account based on the credentials provided through voice input. This module's script designed as such it will prompt user to enter their Gmail username and password and then it will use SMTP server protocol to automate the task for the user and as a result connection will be made.

• Send E-mail through G-mail: - This module will handle the request by user to send email through their g-mail account. The python script for this module will prompt the user to enter their credentials and then it will make connection with their account. After the connection has been done it will further prompt the user to enter the receiver's account e- mail id and it will then allow the user to speak their message and it will repeat it for them and by saying ok it will send the mail. SMTP library in python is used for the above task.

• Read E-mail through G-mail: - This module will handle the request by user to read email through their g-mail account. The python script for this module will prompt the user to enter their credentials and then it will make connection with their account. After the connection has been done it will start fetching the unread mails for the user and will speak it for them with the help of pyttsx3 library in python for text to speech conversion.

## 2.5 System Configuration

### Hardware Requirements

- System: Intel Core i5 2.20 GHz.
- Hard Disk: 500 GB.
- RAM: 2GB
- High Speed Internet
- Speakers or Earphones with Microphone

### Software Requirements

- Operating System: Windows 7 ABOVE.
- Coding Language: Python.
- IDE: Anaconda, PyCharm, etc.

# CHAPTER -3

# LITERATURE OVERVIEW

**International Research Journal of Engineering and Technology (IRJET)**

**Authors** - Naziya Pathan, Nikita Bhoyar, Ushma Lakra, Dileshwari Lilhare

It is estimated that there is a total of 4.92 billion email accounts existing in 2017 and there will be approximately 5.59 billion accounts by the end of 2019. It is also estimated that there is a total of 340.2 million smartphone users in India in the year 2017. This makes emails the most used kind of communication. Novel Based System for Visually Challenged people using Beacon is really challenging. The main goal of this architecture is to provide visually challenged people got know more about the type of conditions they must survive. The prevailing email systems don't give any means of feedback or Talkback service. The most common mail services that we tend to use in our day to day life cannot be used by visually challenged people. This is as a result of they do not offer any facility in order that the person in front will listen the content of the screen. As they cannot visualize what is already present on screen they cannot build out where to click in order to perform the required operations. For a visually impaired person employing a computer system for the first time isn't that convenient as it is for a standard user even though it is user friendly. Though there are several screen readers offered then also these individuals face some minor difficulties. Screen readers speak out whatever content is there on the screen and to perform the actions the person will have to use keyboard shortcuts because mouse location cannot be detected by the screen readers. This means 2 things; one that the user cannot make use of mouse pointer as it is fully inconvenient if the pointer location cannot be derived and second that user should be versed with the keyboard on wherever each key is placed. A user who is new to computer will therefore not use this service as they're not conscious of the key locations. Also, there are some difficulties faced by visually impaired people when using smartphone systems.

**International Journal of Research Studies in Computer Science and Engineering (IJRSCSE) Volume 3, Issue 1, 2016, PP 25-30**

**Authors** - Pranjal Ingle, Harshada Kanade, Arti Lanke

Internet plays a vital role in today's world of communication. Today the world is running based on 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 82 of people living with blindness are 50 aged and above. We must 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 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.

This voice-based email system has come to overcome these problems for visually challenged people. This was made possible with the help of **Python Programming** which offers a lot of features and functionalities. This is automated in such a way that with only voice we can send emails without the need of mouse or keyboard.

## 3.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuations and it has fewer syntactical constructions than other languages.

**Python is interpreted** - Python is processed at runtime my interpreter. Ones does not need to compile the program before executing it. This is similar to PERL and PHP.

**Python is interactive -** you can sit at a python prompt and interact with interpreted directly to write the 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 beginner level programmers and supports the development of a wide range of applications from simple text processing to WWW browser to games.

### 3.1.2 Python Features

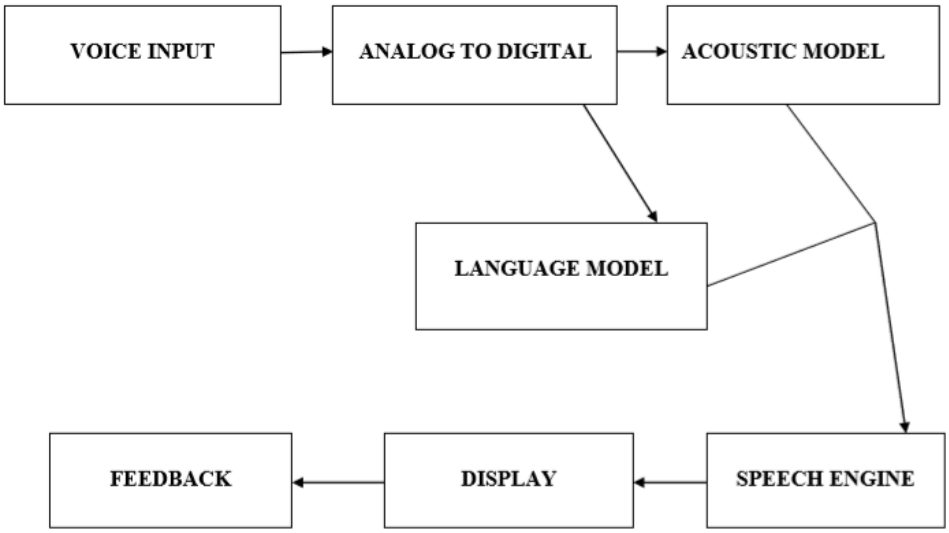
Python's features include –

- **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's code is more clearly defined and visible to the eyes.
- **Easy to maintain** – Python's source code is fairly easy to be maintaining.
- **A broad standard library** – Python has a bulk of libraries which are 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.
- **Databases -** Python provides interfaces to all major commercial databases.
- **Extendable** - you can add low-level module to the python interpreter. These models enable programmers to add or to customize their tools to be more efficient.
- **GUI programming** – Python supports the GUI applications that can be created and ported to many system calls, libraries and window system, 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.

# CHAPTER – 4
# SYSTEM DESIGN

## 4.1 System Architecture

Interaction between different components of the system is shown in the below diagram. Each component sends requests to process information, to and from the system. The process of conversion from speech to text and vice versa is abstracted which is why the user and the receiver components of the system has no idea about their working and so they do not need to interact with those components.




## 4.2 Uml Diagrams

It is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. The UML diagrams are mainly categorized into 9 types. They are:

**Static:**

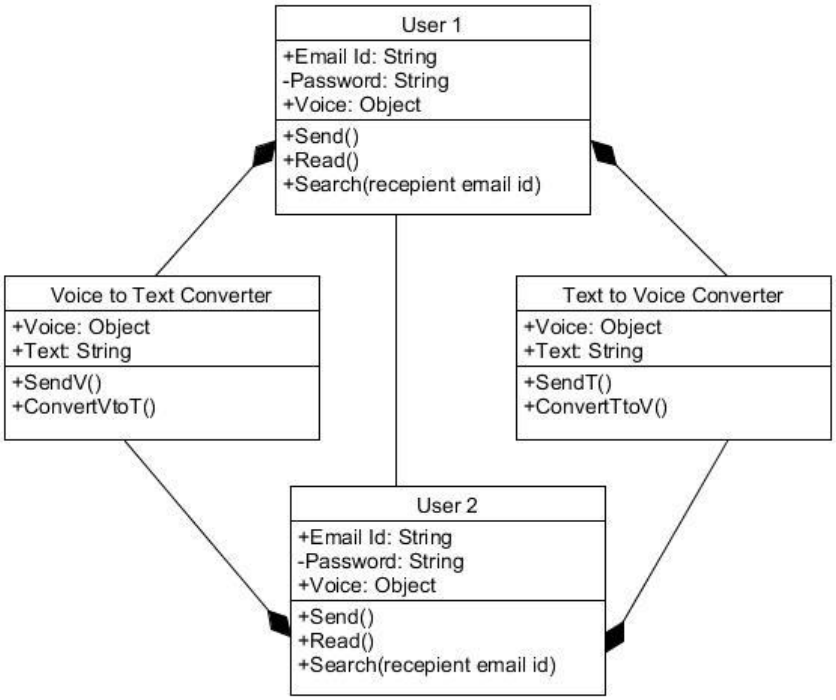
1. Class diagrams
2. Object diagrams
3. Component diagrams
4. Deployment diagrams

**Dynamic:**

1. Use case diagram
2. Sequence diagram
3. Collaboration diagram
4. State chart diagram
5. Activity diagram

## Class Diagram

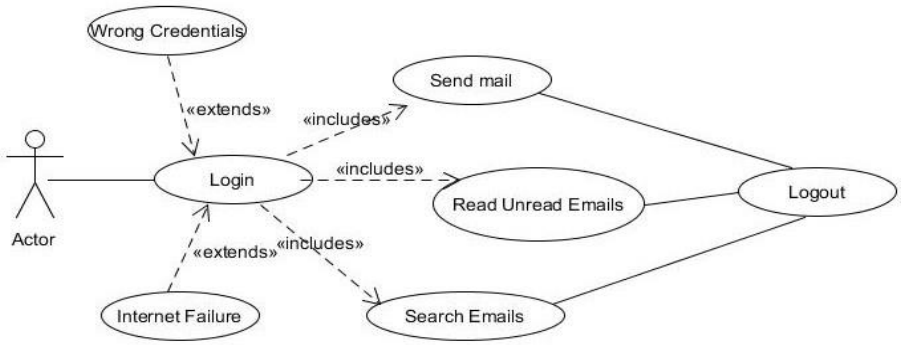
Class Diagram gives the static view of an application. A class diagram describes the types of objects in the system and the different types of relationships that exist among them. It gives an overview of a software system by displaying classes, attributes, operations, and their relationships.




Class diagram to represent the relationship of user to user and converters

## Use case Diagram

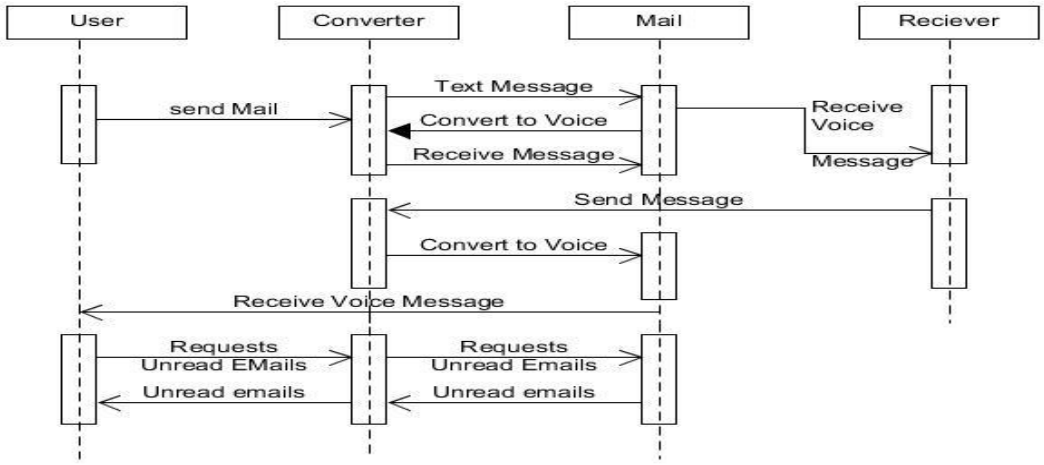
Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly related to design. When a system is analysed to gather its functionalities, use cases are prepared and actors are identified.




Use case diagram for User operations.

## Sequence Diagram

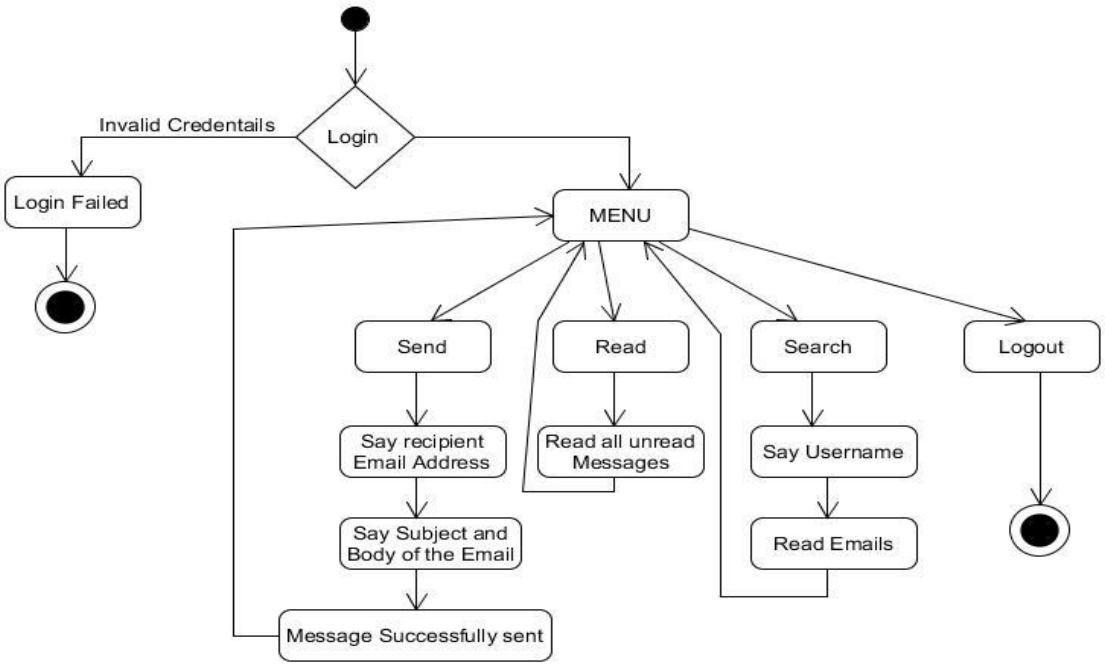
Sequence diagrams are used to capture the dynamic nature but from a different angle. The sequence diagram captures the time sequence of the message flow from one object to another.




Sequence Diagram for user to send and read unread emails.

## Activity Diagram

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.



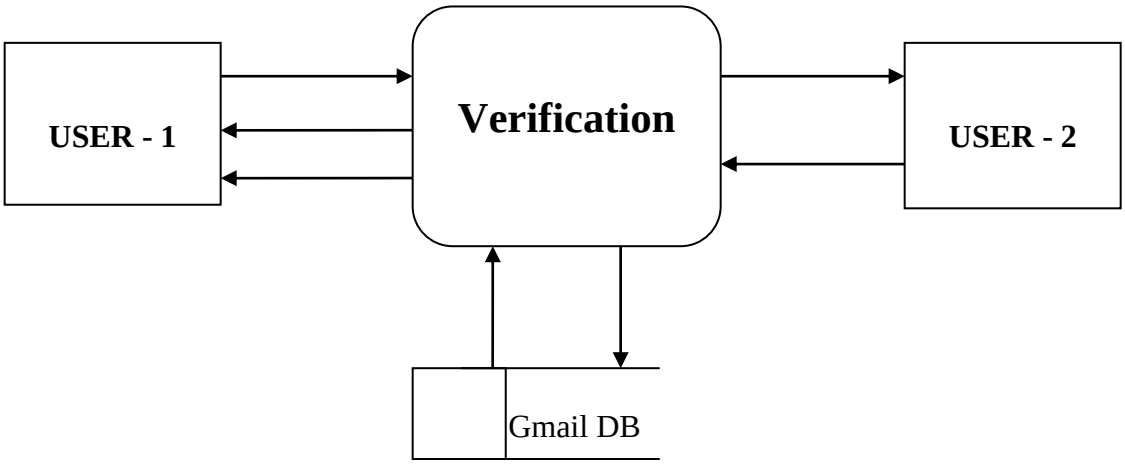

Activity Diagram to represent the flow of project.

## 4.3 SYSTEM DESIGN

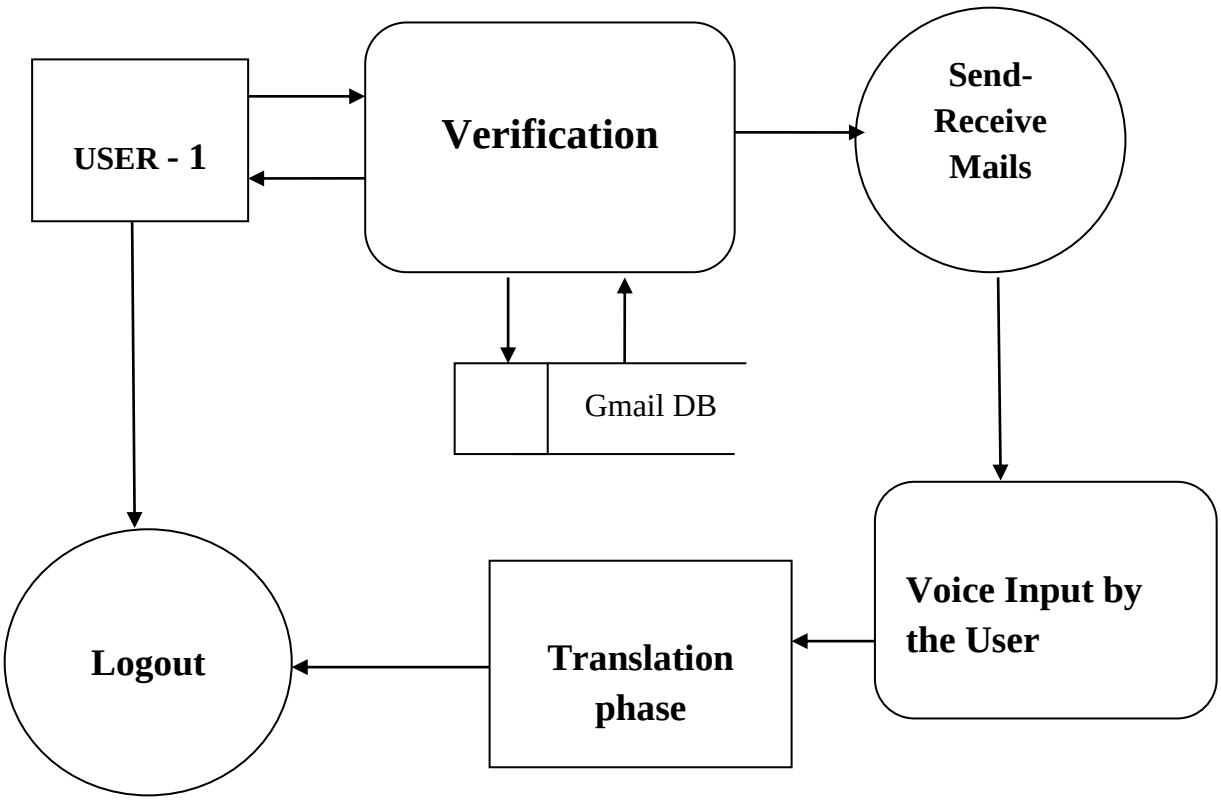
### Data Flow Diagram (DFDs)

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams.

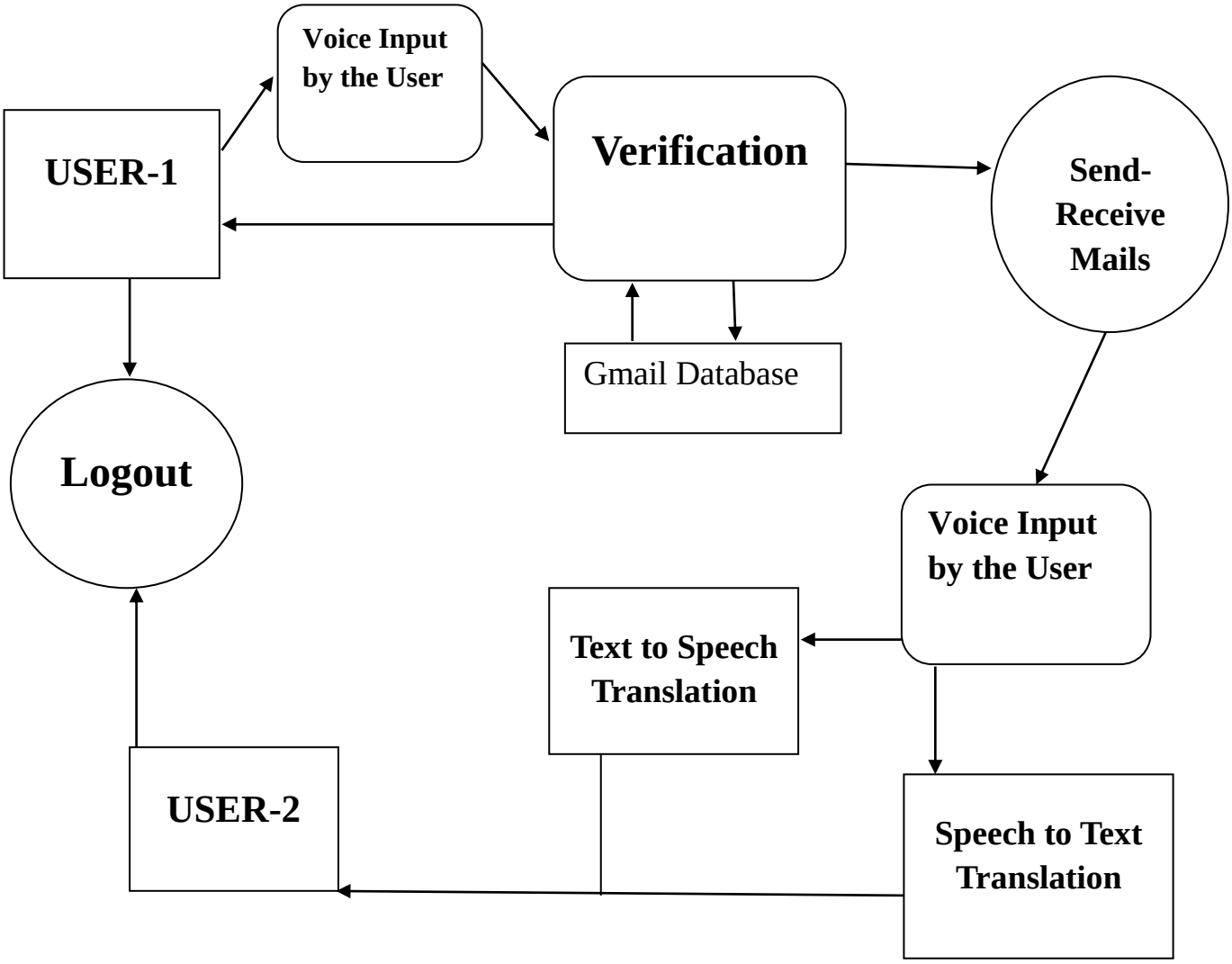
**DFD Level-0**




**DFD Level-1**



**DFD Level-2**


Voice Input by the User
USER-1
Verification
Send-Receive Mails
Gmail Database
Logout
Voice Input by the User
Text to Speech Translation
USER-2
Speech to Text Translation

# CHAPTER - 5

# SAMPLE CODE

```
import sys
import smtplib
import easyimap
import imaplib,re
import speech_recognition as sr
import pyttsx3
from email.mime.text import MIMEText

r = sr.Recognizer()
r.energy_threshold=2500

smtp_ssl_host='smtp.gmail.com'
smtp_ssl_port=465

ch=0
op={'send':1,'read':2,'search':3,'logout':4}

def speakout(label):
    engine=pyttsx3.init()
    engine.say(label)
    engine.runAndWait()
```

```
def verify(text):
    try:
        with sr.Microphone() as source:
            speakout("you have entered")
            speakout(text)
            speakout("Say Yes to continue and No to reenter")
            audio = r.record(source,duration=2)
        text=r.recognize_google(audio)
        return text
    except:
        speakout('Sorry could not recognise Please Try again')
        text=verify(text)
    return text


def voice(label):
    with sr.Microphone() as source:
        speakout(label)
        audio = r.record(source,duration=6)
    text=r.recognize_google(audio)
    text=text.split(' ')
    text=''.join(text)
    msg=text
    print(msg)
```

```
    if(verify(text)=="yes"):
        return msg
    else:
        msg=voice(label)
    return msg

def voicesub(label):
    with sr.Microphone() as source:
        speakout(label)
        audio = r.record(source,duration=5)
    text=r.recognize_google(audio)
    print(text)
    return text

def voicebody(label):
    with sr.Microphone() as source:
        speakout(label)
        audio = r.record(source,duration=15)
    text=r.recognize_google(audio)
    print(text)
    return text


def voicemenu():
```

```
    try:
        with sr.Microphone() as source:
            audio = r.record(source,duration=3)
        text=r.recognize_google(audio)
    except:
        speakout('Sorry could not recognise Please try again')
        text=voicemenu()
    print(text)
    return text


def send(username,password):
    try:
        print('Sending Email')
        sender = username
        print('Enter recipient email address')
        targets = voice('Enter recipient email address')
        print('Enter subject')
        sub=voicesub('Enter subject')
        body=voicebody('Enter the Message to be sent')
        print('Enter the Message to be sent')
        msg = MIMEText(body)
        msg['Subject']=sub
        msg['From']= sender
        server = smtplib.SMTP_SSL(smtp_ssl_host, smtp_ssl_port)
```

```
        server.login(username,password)
        server.sendmail(sender, targets, msg.as_string())
        server.quit()
        speakout('Message successfully Sent')
    except:
        speakout('Message not Sent')


def read(username,password):
    print('Reading out all unread emails')
    i=imaplib.IMAP4_SSL('imap.gmail.com')
    i.login(username,password)
    x,y=i.status('INBOX','(MESSAGES UNSEEN)')
    k='MESSAGES\s+(\d+)'.encode()
    l='UNSEEN\s+(\d+)'.encode()
    mes=int(re.search(k,y[0]).group(1))
    uns=int(re.search(l,y[0]).group(1))
    speakout('you have')
    speakout(uns)
    speakout(' unread emails')
    print("Unread Emails are")
    imapper = easyimap.connect('imap.gmail.com',username, password)
    for mail_id in imapper.unseen(uns):
        speakout('Mail from')
        speakout(mail_id.from_addr)
```

```
        print("Mail from :",mail_id.from_addr)
        speakout('subject')
        speakout(mail_id.title)
        print('Subject :',mail_id.title)
        speakout('Message')
        speakout(mail_id.body)
         print("Message :",mail_id.body)
    if(uns!=0):
        speakout('All unread emails are read')


def search(username,password):
    print('enter a username to search mails')
    search=voice('enter a username to search mails')
    count=0
    imapper = easyimap.connect('imap.gmail.com',username, password)
    for mail_id in imapper.listids(limit=5):
        mail = imapper.mail(mail_id)
        if(mail.from_addr.find(search)!=-1):
            count=count+1
    print(count)
    speakout('You have')
    speakout(count)
    speakout('emails with given username')
    for mail_id in imapper.listids(limit=5):
```

```
        mail = imapper.mail(mail_id)
        if(mail.from_addr.find(search)!=-1):
            speakout('Mail Form')
            speakout(mail.from_addr)
            print("Mail from :",mail.from_addr)
            speakout('Subject')
            speakout(mail.title)
            print('Subject :',mail.title)
            speakout('Message ')
            speakout(mail.body)
            print("Message :",mail.body)


def menu():
    print('say Send to send an email')
    print('say read to check for unread emails')
    print('say Search to search a recepient related email')
    print('say logout to logout and close')
    speakout('say Send to send an email')
    speakout('say read to check for unread emails')
    speakout('say Search to search a recepient related email')
    speakout('say logout to logout and close')
    print("Speakout an Instruction :")
    try:
        z=voicemenu()
```

```
        return op[z]
    except:
        speakout('Sorry could not recognise Please Try again')
        return menu()


#Login Credentials and validation
try:
    username=voice('enter a username')
    password=voice('enter password')
    server = smtplib.SMTP_SSL(smtp_ssl_host, smtp_ssl_port)
    z=server.login(username,password)
    server.quit()
except:
    speakout('Login Failed please try again later')
    sys.exit()
speakout('Login Successful')
print('Login Successful')
while(ch<4):
    ch=menu()
    if(ch==1):
        send(username,password)
    elif(ch==2):
        read(username,password)
    elif(ch==3):
```

```
        search(username,password)
    else:
        speakout('Logout Successful')
        print('Logout Successful')
        break
```

# CHAPTER – 6

# TESTING

## 6.1 TESTING

Testing is the process of evaluating the system or its components with the intent to find out whether it is satisfying the requirements or not testing is executing a system in order to identify any gaps or bugs missing requirements in contrary to actual desired or requirements.

The following are the testing objectives:

- Testing is a process excluding a program with the intent of finding an error.
- Good test has a high probability of finding a yet undiscovered error.
- A successful test is one that uncovers a yet undiscovered error.

## 6.1.2 TESTING METHODOLOGIES

### Integration Testing

Modules integrated by moving down the program design hierarchy. Can use depth first or breadth first, top down integration verifies major control and decision points early in design process. Top-level structure is tested the most.

Top down integration forced by some development tools in programs with graphical user interfaces. Begin construction and testing with atomic modules.

Bottom Up integration testing as its name implies been construction and testing with atomic modules. Because models are integrated from the bottom up, processing required for module subordinate to a given level is always available and the need of Stubs is eliminated.

### System Testing

Once the software product is developed, it is thoroughly tested, and it is delivered to the users. Now, it has been tested by developing it on the screen that is what the given software is comfortable to the environment. The software engineer should consider these issues during

early stages of the software development to release him from the problems which are encountered after completion of the Software. Hence, the tests are conducted to ensure that the software is comfortable with the system, where it is deployed is referred as “System Testing”.

### Unit Testing

This type of testing is performed by developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by respective developers on the individual units of source code assigned areas. The developers use test data that is different from the test date of quality assurance team. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

### Validation Testing

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. It ensures that the product meet the client’s needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on an appropriate environment.

## 6.1.3 DESIGN OF TEST CASES AND SCENARIOS

Test cases involve a set of steps, conditions and inputs that can be used while performing testing tasks. The main intent of this activity is to ensure whether software passes or fails in terms of its functionality and other aspects.

There are many types of test cases such as functional, negative, error, logical test cases, physical test cases, UI test cases, etc. Furthermore, testcases are written to keep track of testing coverage of software. Generally, there are no formal templates that can be used during test case writing.

## Test Case

A test case is a document, which has set of test data preconditions, expected results and post conditions, developed for a test scenario in order to verify compliance against a specific requirement.

Test cases act as a starting point for the test execution and after applying a set of input values that application has definite outcomes and leaves the system at some endpoint or also known as “execution post condition”.

## Test Scenario

It is one-line statement that notifies what area in the application will be tested. Testing scenarios are used to ensure that all process flows are tested from end to end. A particular area of an application can have as little as one test scenario to a few hundred scenarios depending on the magnitude and complexity of the application.

The terms ‘Test Scenario’ and ‘Test-cases’ are used interchangeably, however a test scenario has several steps, where as a test case a single step. Viewed from this perspective, test scenarios are test cases, but they include several test cases and the sequence that they should be executed. Apart from this, each test cases dependent on output from the previous test case.

## 6.2 Test Cases

| **Test Case ID** #1 | | **Test Case Description:** Logging in into our account | | |
|---|---|---|---|---|
| **S#** | **Prerequisites** | **S#** | **Test Data Requirement** | |
| 1 | username and password | 1 | All required packages to be installed. | |
| **Test Condition** | | | | |
| Entering the Correct Username and Password of a Gmail Account. | | | | |
| **Step#** | **Step Details** | **Expected Results** | **Actual Results** | **Pass/Fail/Not Executed/ Suspended** |
| 1 | Enter username and password Correctly | Login Successful and Connection establishes with Gmail server. | Login Successful and Connection establishes with Gmail server. | Pass |
| 2 | Enter username or password Incorrectly | Login Unsuccessful and Connection does not establish with Gmail server. | Login Unsuccessful and Connection does not establish with Gmail server. | Pass |
| 3 | Enter username and password Correctly but No Internet Connectivity. | Login Unsuccessful and Connection does not establish with Gmail server. | Login Unsuccessful and Connection does not establish with Gmail server. | Pass |

Table 1: Logging in into our account

| **Test Case ID** #2 | | **Test Case Description:** Sending a mail | | |
|---|---|---|---|---|
| **S#** | **Prerequisites** | **S#** | **Test Data Requirement** | |
| 1 | Login Successfully done | 1 | All required packages to be installed. | |
| **Test Condition** | | | | |
| Sending mails from our Gmail account.<br>Asking for recipient email address. | | | | |
| **Step#** | **Step Details** | **Expected Results** | **Actual Results** | **Pass/Fail/Not Executed/ Suspended** |
| 1 | Asking recipient email address | Asks recipient email address through Voice. | Asks recipient email address through Voice. | Pass |
| 2 | Confirmation of recipient email address | Asks recipient email address through Voice again and tells user to say yes or no. | Asks recipient email address through Voice again and tells user to say yes or no. | Pass |
| 3 | Asking to say Subject and body of the mail | Asks the user to say subject and body od the mail through Voice. | Asks the user to say subject and body of the mail through Voice. | Pass |

Table 2: Sending a mail

| **Test Case ID** #3 | | **Test Case Description:** Unread mails | | |
|---|---|---|---|---|
| **S#** | **Prerequisites** | **S#** | **Test Data Requirement** | |
| 1 | Login Successfully done | 1 | All required packages to be installed. | |
| **Test Condition** | | | | |
| Checking for unread mails in our Gmail account. | | | | |
| **Step#** | **Step Details** | **Expected Results** | **Actual Results** | **Pass/Fail/Not Executed/ Suspended** |
| 1 | Checking and retrieving the object of mail. | Mail Object is stored in a variable. | Mail Object is stored in a variable. | Pass |
| 2 | Counting the number of unread emails. | Displays the number of unread emails to the user. | Displays the number of unread emails to the user. | Pass |
| 3 | Traversing through each unread email. | Displaying from address, subject, body of each mail. | Displaying from address, subject, body of each mail. | Pass |

Table 3: Unread mails

| **Test Case ID** #4 | | **Test Case Description:** searching mails | | |
|---|---|---|---|---|
| **S#** | **Prerequisites** | **S#** | **Test Data Requirement** | |
| 1 | Login Successfully done | 1 | All required packages to be installed. | |
| **Test Condition** | | | | |
| searching for particular username mails in our Gmail account. | | | | |
| **Step#** | **Step Details** | **Expected Results** | **Actual Results** | **Pass/Fail/Not Executed/ Suspended** |
| 1 | Asking email address to search | Asks email address through Voice. | Asks email address through Voice. | Pass |
| 2 | Confirmation of recipient email address | Asks recipient email address through Voice again and tells user to say yes or no. | Asks recipient email address through Voice again and tells user to say yes or no. | Pass |
| 3 | Counting the number of emails present. | Displays the number of emails present to the user. | Displays the number of emails present to the user. | Pass |
| 4 | Traversing through each email. | Displaying from address, subject, body of each mail. | Displaying from address, subject, body of each mail. | Pass |

Table 4: Searching mails

# CHAPTER -7

# OUTPUT SCREENS

**Login Interface:**

If the username and Password are correct, then

```
Say username
miniproject2411@gmail.com
Say your password
[redacted]
Login Successful
Menu :
say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
Speakout an Instruction :
```

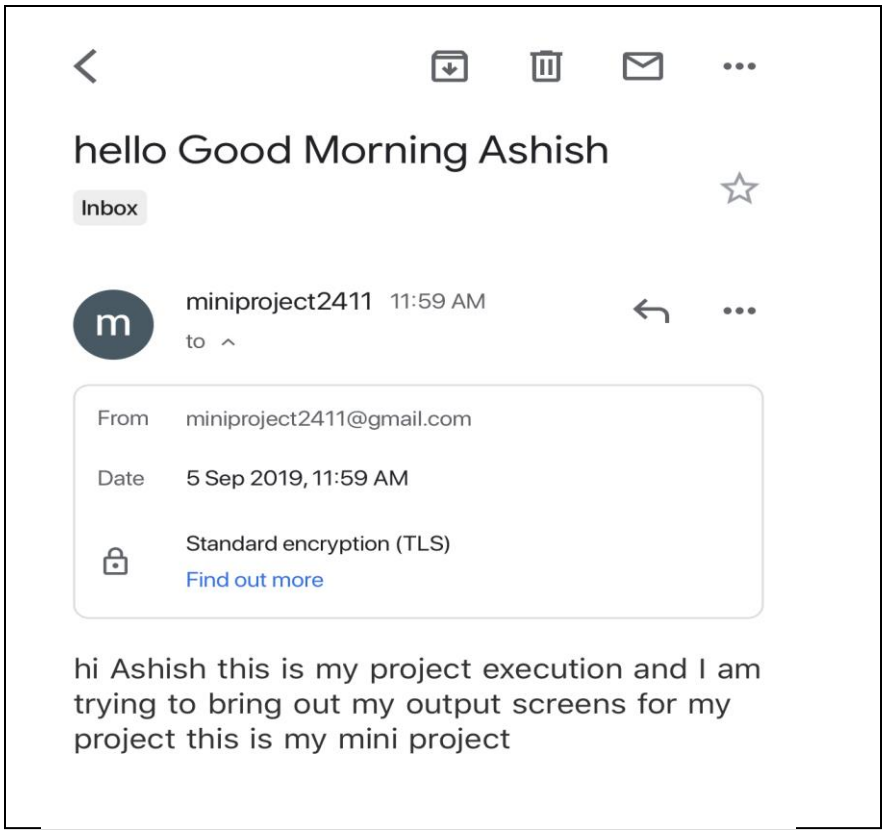
**If the username or password are incorrect, then**

```
Say username
miniproject2411@gmail.com
Say your password
90004676
Login Failed
Due to Wrong Credentails or Network Failure
Try Again Later
```

## Sending a mail:

```
say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
Speakout an Instruction :
send
Sending Email
Enter recipient email address
miniproject9999@gmail.com
Enter subject
hello Good Morning Ashish
hi Ashish this is my project execution and I am trying to bring out my output screens
 for my project this is my mini project
Enter the Message to be sent
say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
```

## Shapshot of mail received from the receiver's mail account

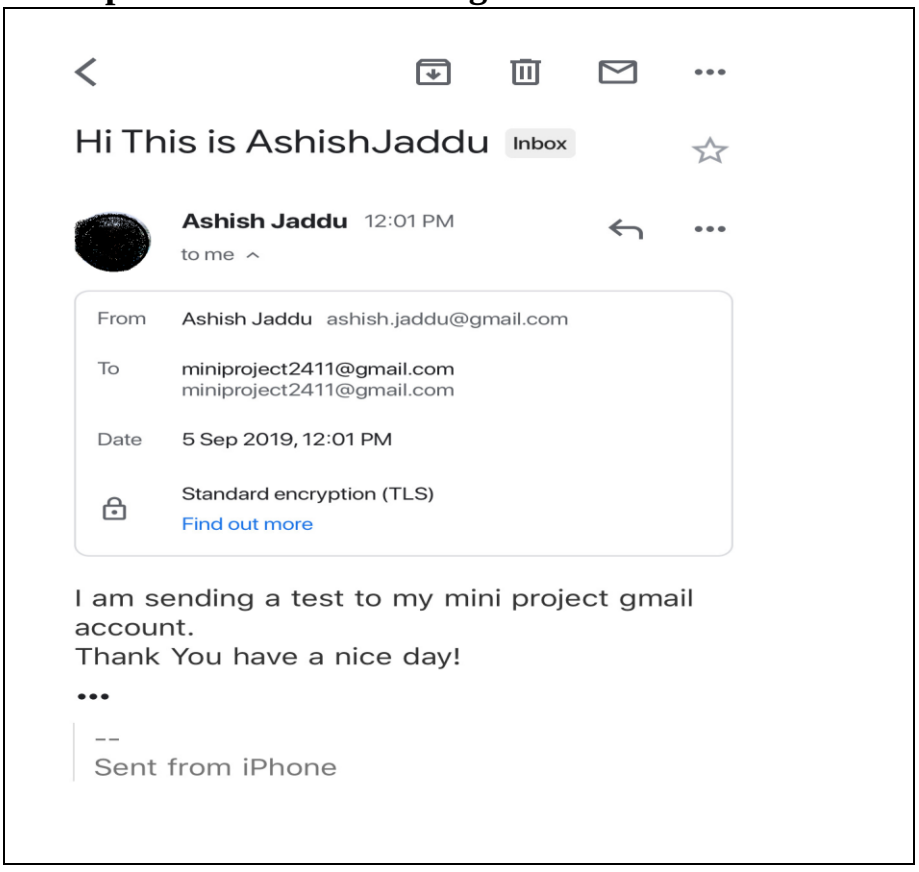


**Reading unread emails:**

```
say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
Speakout an Instruction :
read
Reading out all unread emails
Unread Emails are
Mail from : Ashish Jaddu <ashish.jaddu@gmail.com>
Subject : Hi This is AshishJaddu
Message : I am sending a test to my mini project gmail account.
Thank You have a nice day!
--
Sent from iPhone

say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
```

**Shapshot of mail from the gmail account:**



## Searching Emails:

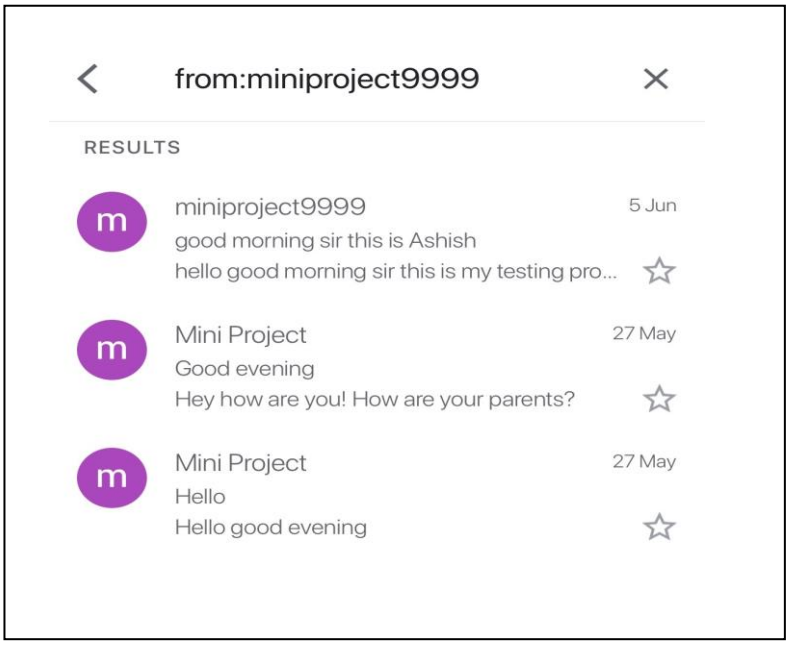
```
Login Successful
say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
Speakout an Instruction :
search
enter a username to search mails
miniproject9999@gmail.com
3
You have 3 emails with given username
Mail from : miniproject9999@gmail.com
Subject : good morning sir this is Ashish
Message : hello good morning sir this is my testing process of my Python programming

Mail from : Mini Project <miniproject9999@gmail.com>
Subject : Hello
Message : Hello good evening

Mail from : Mini Project <miniproject9999@gmail.com>
Subject : Good evening
Message : Hey how are you!
How are your parents?

say Send to send an email
say read to check for unread emails
say Search to search a recepient related email
say logout to logout and close
```

## Shapshot of mail from the gmail account:



# CHAPTER - 8

# CONCLUSION

## 8.1 Conclusions

It is basically designed to help handicapped people, who face difficulties in accessing the computer system.

Voice mail architecture helps blind people to access e-mail. It basically describes the voice mail architecture that can be used by blind people to access their e-mails. This architecture will also reduce cognitive load taken by blind to remember and type characters using the keyboard. Our application will help physically challenged people to access the world according to their ability.

In future this application can be enhanced and will not only can be implemented for email services but also it can be useful to other services like texting, making notes, operating other application through voice.

## 8.2 Further Enhancements

For the further development of the application, the attachments like images, word documents, audio and video files can be incorporated. Encryption and decryption algorithm can used to protect the username and password that is passed during login. More commands can be used to for different operations like search, mark important, delete, archive, go back, report spam, forward. Also, the development of logging into the account without telling the password outside.

# CHAPTER – 9

# BIBILIOGRAPHY

## 9.1 Books References


- Learning Python by Mark Lutz
- Python Crash Course by Eric Matthews
- Python 3 by Priya Sen
- Artificial Intelligence with Python by Prateek Joshi


## 9.2 Websites References


- https://www.python.org
- https://realpython.com/python-speech-recognition
- https://docs.python.org/2/library/smtplib.html
- https://www.tutorialspoint.com/python/python_sending_email.htm
- https://pypi.org/project/SpeechRecognition/


## 9.3Technical Publication References


- International Research Journal of Engineering and Technology (IRJET) Authors - Naziya Pathan, Nikita Bhoyar, Ushma Lakra, Dileshwari Lilhare
- International Journal of Research Studies in Computer Science and Engineering (IJRSCSE) Volume 3, Issue 1, 2016, PP 25-30
  Authors - Pranjal Ingle, Harshada Kanade, Arti Lanke

# CHAPTER – 10

# APPENDICES

## 10.1 Python Application

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It's high-level built in data structures combined with dynamic typing and dynamic binding make it attractive for development as well as for use as 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.

## 10.2 Python Libraries

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

Speech Recognition

Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text.

Smtplib

Simple Mail Transfer Protocol (SMTP) is a protocol, which handles sending e-mail and routing e-mail between mail servers.

Python provides smtplib module, which defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon.

Email.Mime

Python's Multipurpose Internet Mail Extensions (MIME) is an Internet standard that extends the format of email to support:

- Text in character sets other than ASCII
- Non-text attachments: audio, video, images, application programs etc.
- Message bodies with multiple parts
- Header information in non-ASCII character sets

Pyttsx3

pyttsx is a cross-platform text to speech library which is platform independent. The major advantage of using this library for text-to-speech conversion is that it works offline. It is a Python package supporting common text-to-speech engines on Mac OS X, Windows, and Linux.

Imaplib

Imaplib implements a client for communicating with Internet Message Access Protocol (IMAP) version 4 servers. The IMAP protocol defines a set of commands sent to the server and the responses delivered back to the client. Most of the commands are available as methods of the IMAP4 object used to communicate with the server.

## 10.3PYCHARM

**Overview of Important Features and Tools Provided by PyCharm**

**Code Editor**

The intelligent code editor provided by PyCharm enables programmers to write high-quality Python code. The editor enables programmers to read code easily through color schemes, insert Indents on new lines automatically, pick the appropriate coding style and avail context-aware code completion Suggestions. At the same time the programmers can also use the editor to expand a code block to an expansion or logical block avail code snippets, format the code base, identifies errors and misspelling, detect duplicate and auto generate code.

**Code Navigator**

The smart code navigator options provided by PyCharm help programmers to edit and improve code without putting extra time and effort. The IDE makes it easier for programmers to go to a class, file and symbols, along with the go to declarations invoked from a reference. The user can even find an item in the source code, code snippets, UI elements or user actions almost immediately. They can further locate usage of various symbols and set bookmarks in the code. At the same time the developers can even take the advantage of code navigation feature to scrutinize the code thoroughly in the lens mode.

**Refactoring**

PyCharm Makes it easier for developers to implement both local and global changes quickly and efficiently. The developers can take advantage of the refactoring options provided by IDE while writing plain Python code and working with python frameworks. They can avail the rename and move refactoring for files, classes, functions, method, properties, parameters and local/global variables. Likewise, they can improve code quality by extracting variables, fields, constant and parameters. Also, PyCharm allows programmers to break up longer classes and methods through extract method.

**Database tools**

In addition to supporting various python libraries and frameworks. PyCharm allows developers to work with several relational databases including Oracle, SQL Server, MySQL, PostgreSQL. The developers can further use the IDE to run queries edit SQL code, browse data, altar table data, altar/analyses schemas. PyCharm further support SQLALCHEMY library and inject SQL code into code written in various programming languages. The professional addition of IDE further makes it easier for developers to handle large volumes of data efficiently through data grids.

**Software Testing**

Like other IDEs, PyCharm also comes with features and tools to simplify Python application testing. It allows developers to perform unit testing through popular python testing frameworks like Nose, Attest and Doctests. The testers even have option to run individual or multiple test files and test cases. They can further integrate the IDE with coverage.py to measure code coverage while testing applications. While testing multi-threaded applications, the testers can use the thread concurrency visualization option options provided by IDE to control application fully and efficiently. At the same time PyCharm enables users to deliver high-quality software by implementing behavior driven development (BDD).

## SETTING UP PYCHARM

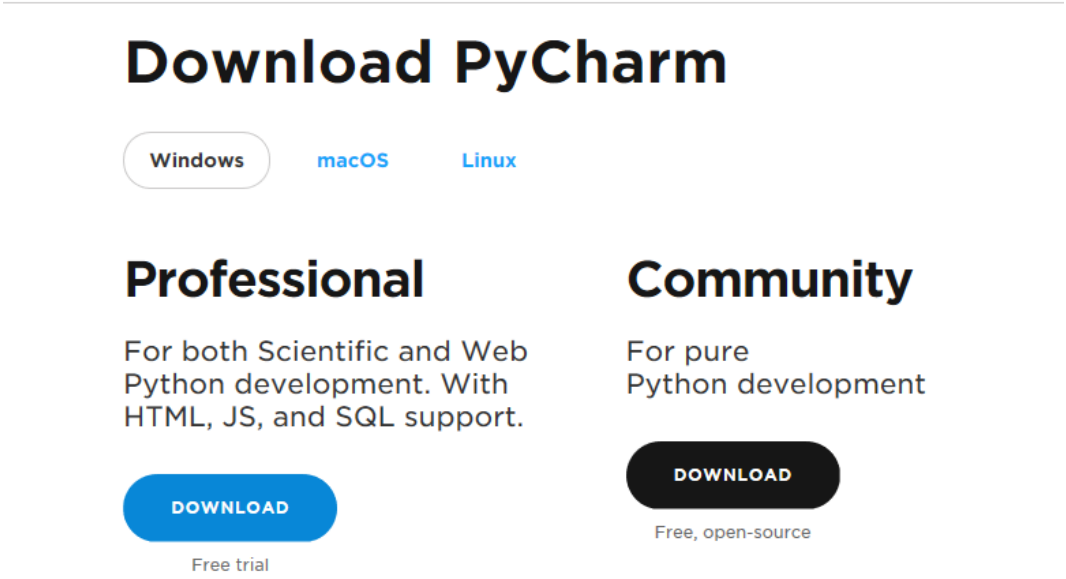
## Installation Steps:

1. To get the installer, first you must make a **JetBrains Educational Account**. (Note: you will have to confirm your account through your email)

2. Go to JetBrains and find **PyCharm**.

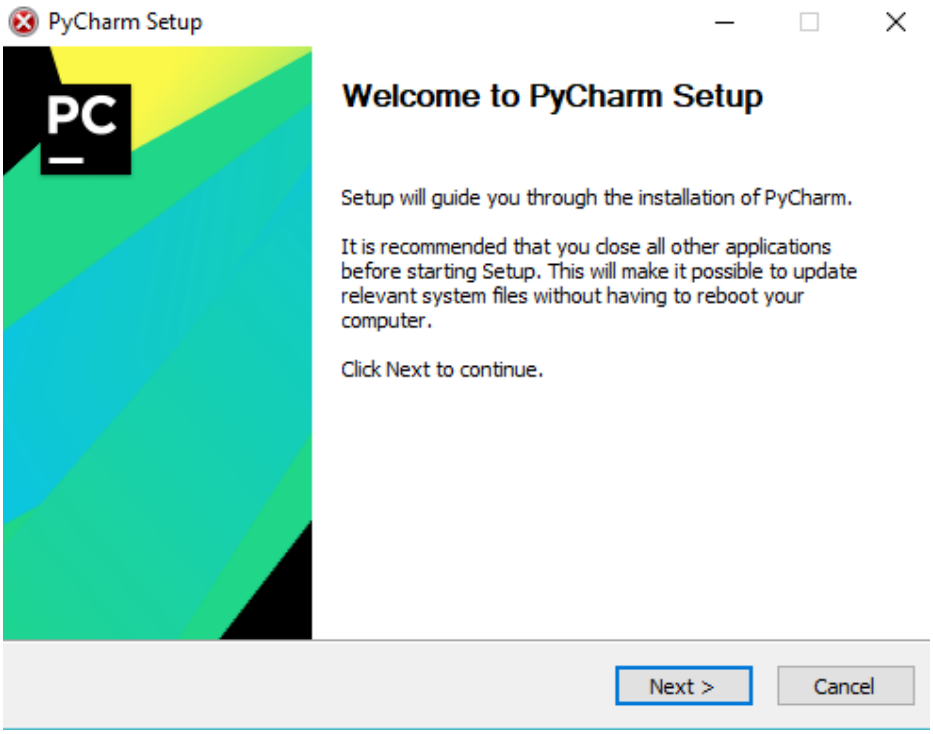
3. Click the **Download** button found in the middle of the page or in the top right corner of the screen.


4. After Clicking download, the website will ask what platform and whether you wish to download professional or community option. Select the desired operating system and click **Professional**.

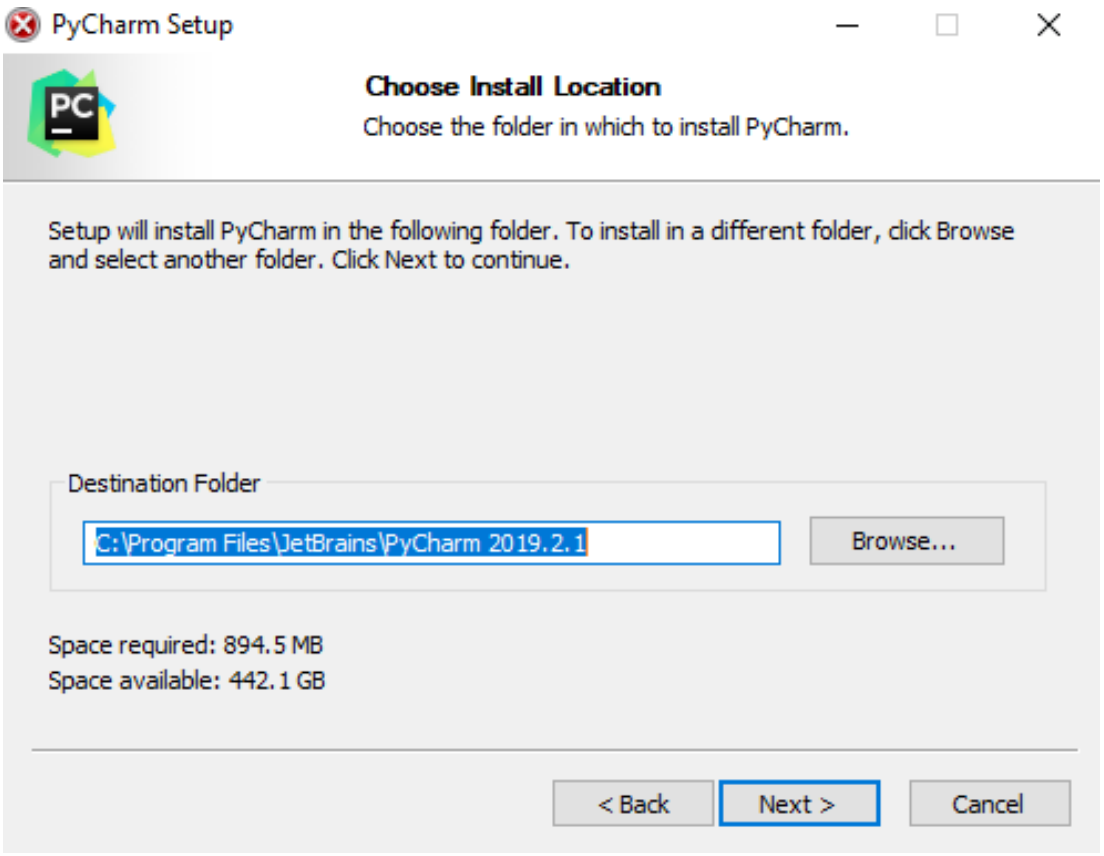



5. After clicking the button, a browser will appear in the **bottom left** corner of the screen..When its done downloading, click the button.If the button does not appear,then navigate to your downloads folder and open **PyCharm** setup file.

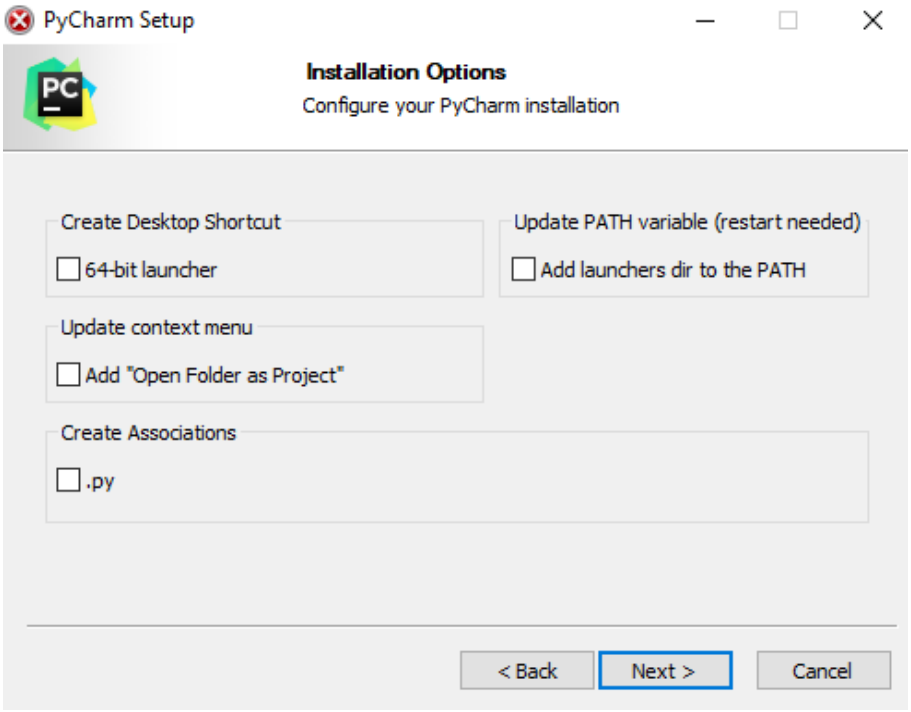
6. The Setup Wizard should open up. Click **Next** to Continue.



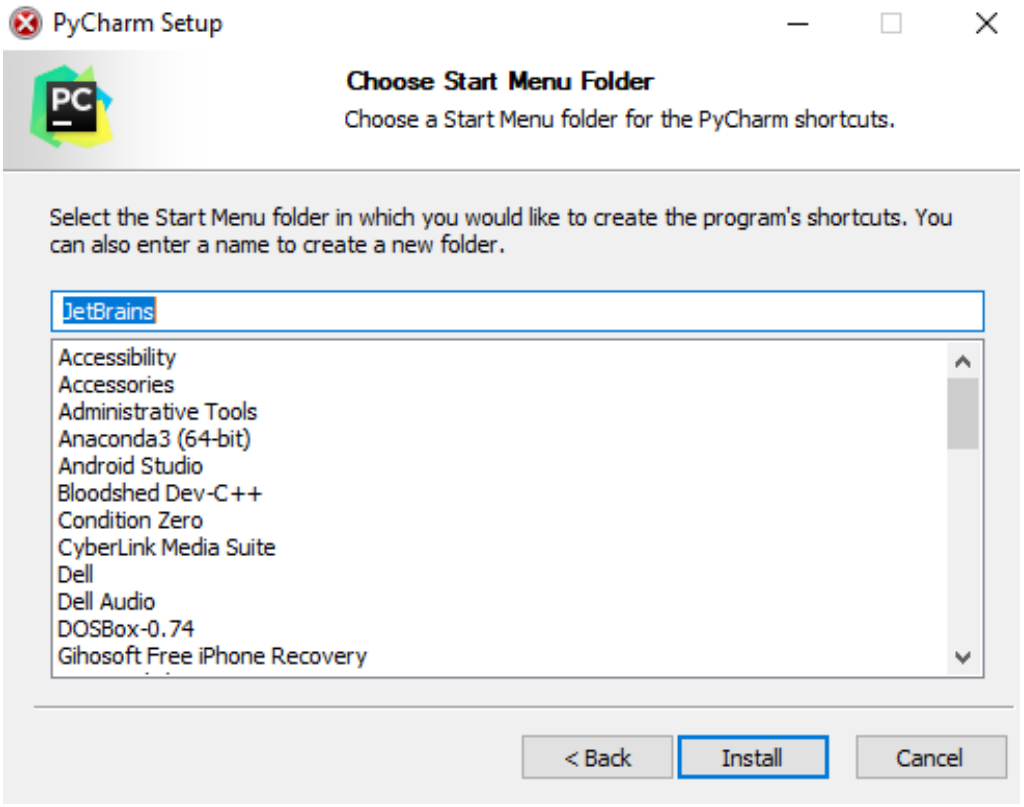

7. The program will ask for an install location. Click **Browser** if you wish to change the default location.Click **Next** to Continue.



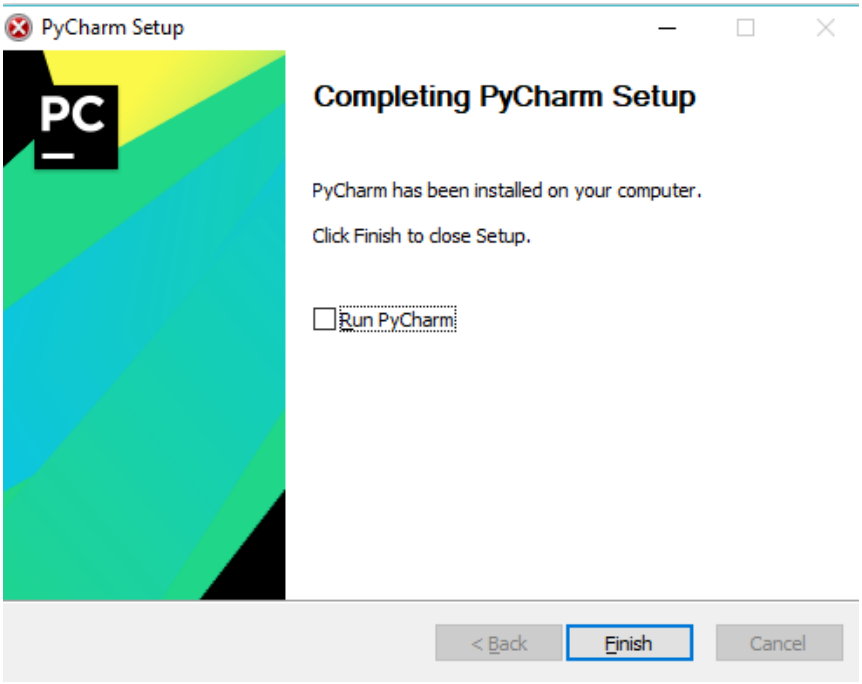

8. Installation options available are 64-bit launcher and few others. Check them as per the requirement. Click Next to continue. (By default, they are unchecked).



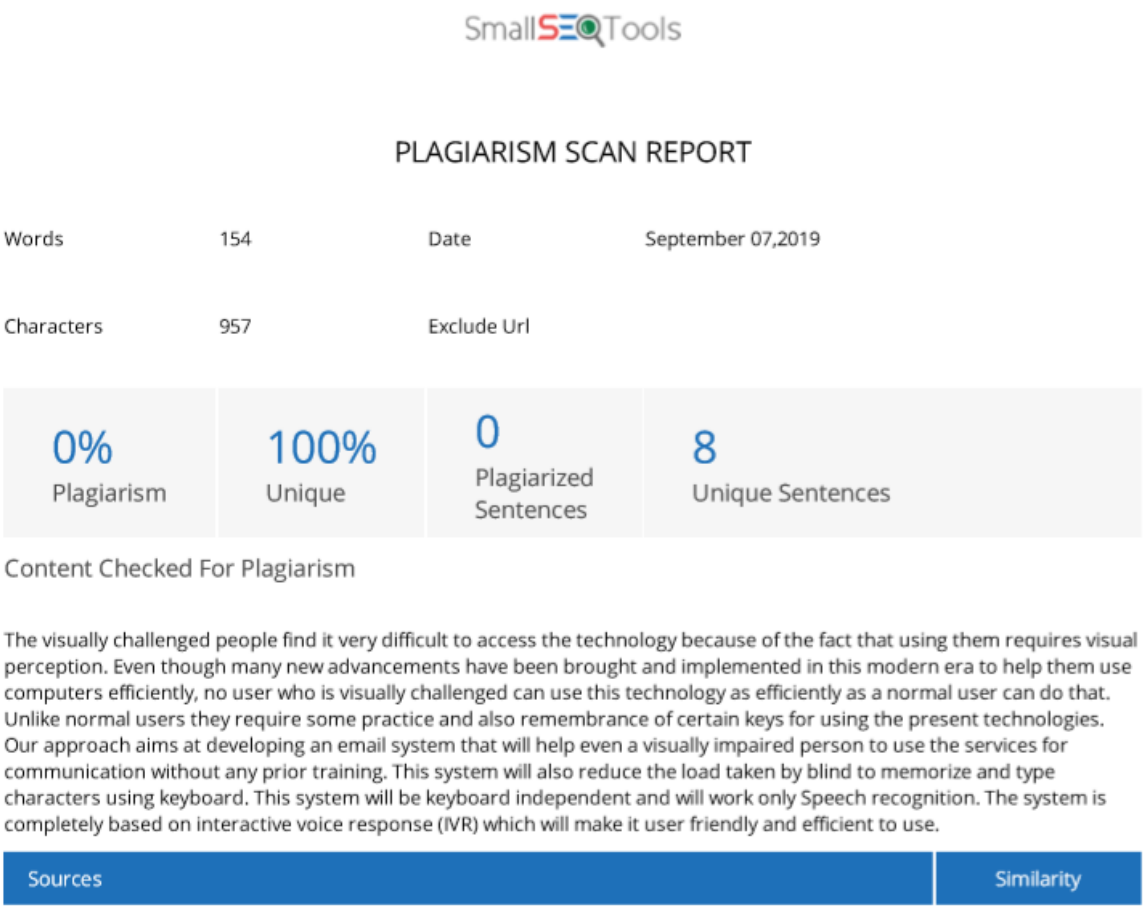

9. Choose a Start Menu Folder. Keep JetBrains selected and click Install.



10. When the program is done Installing. Click **Finish.**



# Plagiarism Report



SmallSEOTools

PLAGIARISM SCAN REPORT

| | | | |
|---|---|---|---|
| Words | 154 | Date | September 07,2019 |
| Characters | 957 | Exclude Url | |

| 0% | 100% | 0 | 8 |
|---|---|---|---|
| Plagiarism | Unique | Plagiarized Sentences | Unique Sentences |

Content Checked For Plagiarism

The visually challenged people find it very difficult to access the technology because of the fact that using them requires visual perception. Even though many new advancements have been brought and implemented in this modern era to help them use computers efficiently, no user who is visually challenged can use this technology as efficiently as a normal user can do that. Unlike normal users they require some practice and also remembrance of certain keys for using the present technologies. Our approach aims at developing an email system that will help even a visually impaired person to use the services for communication without any prior training. This system will also reduce the load taken by blind to memorize and type characters using keyboard. This system will be keyboard independent and will work only Speech recognition. The system is completely based on interactive voice response (IVR) which will make it user friendly and efficient to use.

| Sources | Similarity |
|---|---|