A PROJECT REPORT

ON

JINGLE

(A GUI ENHANCED CHAT BOT)

*Under the Guidance Of*

**Ms. NAMITA AWASTHI**

*By*

**KHUSHI GUPTA**

**Roll no.:0302388**

**BACHELOR OF COMPUTER APPLICATION**

**(Session: 2022-23)**

**ALLENHOUSE BUSINESS SCHOOL**

**ROOMA, KANPUR**



**TABLE OF CONTENTS** **PAGE**

**CERTIFICATE............................................................................................................................ I**

**ACKNOWLEDGEMENT...........................................................................................................II**

**DECLARATION........................................................................................................................ III**

**1.INTRODUCTION.................................................................................................................... IV**

**2.OBJECTIVE............................................................................................................................. V**

**3.FRONT END TECHNOLOGY USED...................................................................................VI**

3.1 PYTHON

**4.BACK END TECHNOLOGY USED.....................................................................................VII**

4.1 MYSQL

**5.SOFTWARE AND HARDWARE REQUIREMENT SPECIFICATION....................................................................................................................... VIII**

5.1 HARDWARE REQUIREMENTS

5.2 SOFTWARE REQUIREMENTS

**6.MODULES USED...................................................................................................................... IX**

**7.MODULE DESCRIPTION....................................................................................................... IX**

**8.DATA FLOW DIAGRAM......................................................................................................... X**

**9.ENTITY RELATIONSHIP DIAGRAM................................................................................... XI**

**10.DATABSE DESIGN.................................................................................................................XII**

**11.DATABSE TABLE................................................................................................................. XIII**

**12.CODE….................................................................................................................................XIV-XVII**

**13.TESTING........................................................................................................................... XVIII-XIX**

13.1 UNIT TESTING

13.2 INTEGRATION TESTING

13.3 SYSTEM TESTING

13.4 USER ACCEPTANCE TESTING

13.5 VALIDATION

13.6 OUTPUT TESTING

**14.SCREENSHOTS..................................................................................................................... XX-XXI**

**15.FUTURE SCOPE OF THE PROJECT................................................................................. XXII**

**16.CONCLUSION........................................................................................................................ XXIII**

**17.BIBLIOGRAPHY................................................................................................................... XXIV**



**CERTIFICATE**

This is to certify that **KHUSHI GUPTA, XXXXX(123456))** have successfully completed her project entitled **JINGLE,** is a record of their own work carried out for partial fulfillment of the requirements for the fifth semester minor project training in **Bachelor of Computer Application (BCA)** and prepare this report under my guidance and supervision as required for the completion of their degree.

**She possesses a good moral character to the best of my knowledge and belief. We wish her success in life.**

**Date:\_\_\_\_\_\_\_\_**

**Project Coordinator**

**Ms. Namita Singh**

**Place:\_\_\_\_\_\_\_\_**



**ACKNOWLEDGEMENT**

I feel an immense pleasure expressing my thanks to those who have contributed their valuable time and resources in helping to achieve success in this project and project report.

I would like to sincerely thank and expresses my gratitude to **Ms. NAMITA AWASTHI (BCA)** for her continuous encouragement, guidance and help in completion of our project training.

A special thanks to **all faculty members** for their kind co-operation and moral support without which the making of this project would have been impossible.

**KHUSHI GUPTA**

**BCA 5th semester**



**DECLARATION**

I hereby declare that the work which is been presented in the project report entitled **PROJECT REPORT on JINGLE** in partial fulfillment of the degree of **Bachelor of Computer Application** is an authentic record of my work carried out under the essential guidance of Project coordinator **Ms. NAMITA AWASTHI.** The work has been carried out at **ALLENHOUSE BUSINESS SCHOOL.**

**BY: KHUSHI GUPTA SIGNATURE:**

**INTRODUCTION**

**JINGLE is a GUI enhanced CHAT BOT.**

The idea behind jingle is to provide a friendly assistance to the user for sending mails on time and to avoid pending mail scenarios.

JINGLE carries a one to one conversation with the user. It helps the user to send mail to the desired recipient with a desired subject and body.

Jingle has the capability recognize the recipient’s email address simply by calling out their nicknames, which makes it more user friendly and convenient.

Not only this, Jingle also keeps a record of its sent mails. Which helps the user to keep a track of the work accomplished, as well acts as a proof for any future references.

Jingle uses Gmail as an email service provider for from sender’s end for sending mails to the recipient of Gmail or any other email service provider. Although the email service provider can be modified according to user’s requirement.

**OBJECTIVE**

Currently we all our aware of the vast use of technology in our day to day life. Probably every possible activity is shifting to our phone or computer screens, from bill payments to booking movie tickets, from ordering food to selling old items, from public sector to private sector everyone is adapting to this technical revolution.

Jingle is a technical revolution for the assistance people are looking for in their day to day lives. In a very hectic and stressful schedule filled with technology and lack of human touch, jingle provides a way to get this human touch without being outdated i.e. a humanlike tech application which will be as accurate as a computer but will also be as polite as an actual human assistant.

Jingle helps the user to avoid procrastination for sending mails in time as its user friendly approach makes it very easy for the user to keep their mailing queue settled.

**FRONT END TECHOLOGY USED IN THE PROJECT**

**Python**

**Python** is a [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Its design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with the use of [significant indentation](https://en.wikipedia.org/wiki/Off-side_rule).

Python is [dynamically-typed](https://en.wikipedia.org/wiki/Type_system#DYNAMIC) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly [procedural](https://en.wikipedia.org/wiki/Procedural_programming)), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [functional programming](https://en.wikipedia.org/wiki/Functional_programming). It is often described as a "batteries included" language due to its comprehensive [standard libra](https://en.wikipedia.org/wiki/Standard_library)ry.

### Why Python?

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-oriented way or a functional way.

### Python Syntax compared to other programming languages

* Python was designed for readability, and has some similarities to the English language with influence from mathematics.
* Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
* Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

**BACK END TECHNOLOGY OF THE PROJECT**

**MySQL**

**MySQL** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). Its name is a combination of "My", the name of co-founder [Michael Widenius](https://en.wikipedia.org/wiki/Michael_Widenius)'s daughter My, and "SQL", the abbreviation for [Structured Query Language](https://en.wikipedia.org/wiki/Structured_Query_Language). A [relational database](https://en.wikipedia.org/wiki/Relational_database) organizes data into one or more data tables in which data may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an [operating system](https://en.wikipedia.org/wiki/Operating_system) to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software) under the terms of the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), and is also available under a variety of [proprietary](https://en.wikipedia.org/wiki/Proprietary_software) licenses. MySQL was owned and sponsored by the [Swedish](https://en.wikipedia.org/wiki/Sweden) company [MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB), which was bought by [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) (now [Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation)). In 2010, when [Oracle acquired Sun](https://en.wikipedia.org/wiki/Acquisition_of_Sun_Microsystems_by_Oracle_Corporation), Widenius [forked](https://en.wikipedia.org/wiki/Fork_(software_development)) the [open-source](https://en.wikipedia.org/wiki/Open-source) MySQL project to create [MariaDB](https://en.wikipedia.org/wiki/MariaDB).

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) [web application](https://en.wikipedia.org/wiki/Web_application) [software stack](https://en.wikipedia.org/wiki/Software_stack) (and [others](https://en.wikipedia.org/wiki/List_of_AMP_packages)), which is an acronym for [*Linux*](https://en.wikipedia.org/wiki/Linux)*,*[*Apache*](https://en.wikipedia.org/wiki/Apache_HTTP_Server)*, MySQL,*[*Perl*](https://en.wikipedia.org/wiki/Perl)*/*[*PHP*](https://en.wikipedia.org/wiki/PHP)*/*[*Python*](https://en.wikipedia.org/wiki/Python_(programming_language)). MySQL is used by many database-driven web applications, including [Drupal](https://en.wikipedia.org/wiki/Drupal), [Joomla](https://en.wikipedia.org/wiki/Joomla), [phpBB](https://en.wikipedia.org/wiki/PhpBB), and [WordPress](https://en.wikipedia.org/wiki/WordPress). MySQL is also used by many popular [websites](https://en.wikipedia.org/wiki/Website), including [Facebook](https://en.wikipedia.org/wiki/Facebook), [Flickr](https://en.wikipedia.org/wiki/Flickr), [MediaWiki](https://en.wikipedia.org/wiki/MediaWiki), [Twitter](https://en.wikipedia.org/wiki/Twitter), and [YouTube](https://en.wikipedia.org/wiki/YouTube).

**SOFTWARE AND HARDWARE REQUIREMENT**

**SPECIFICATION**

**Hardware Requirements**

**Processor** : Intel

**RAM** : 4GB OR MORE

**Monitor**  : CRT or LCD monitor

**Software Requirements**

**Front End** : PYTHON

**Language** : PYTHON

**Back End** : MySQL

**Operation System**  : Window XP or above

**Browser** : Any latest browser

**MODULES USED IN PROJECT**

import smtplib

import speech\_recognition as sr

import pyttsx3

from email.message import EmailMessage

from tkinter import \*

from tkinter import ttk

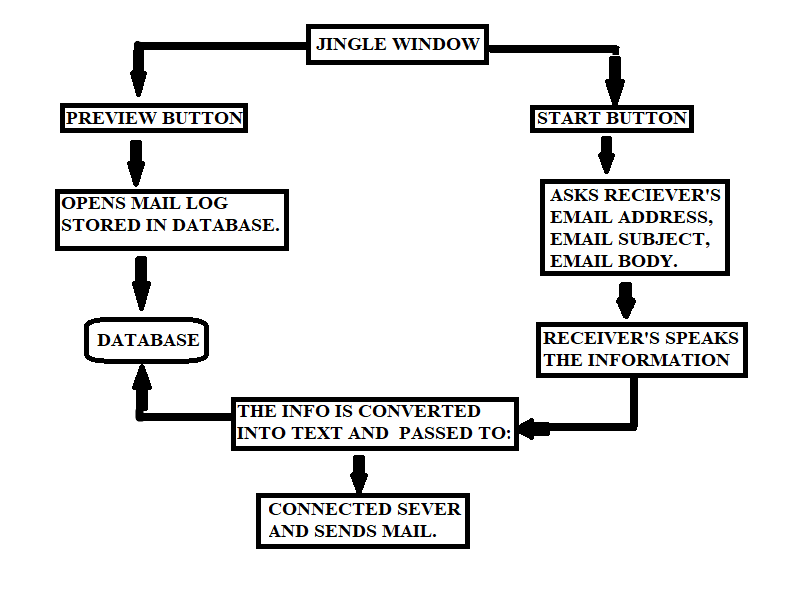
from datetime import datetime

import mysql.connector as msl

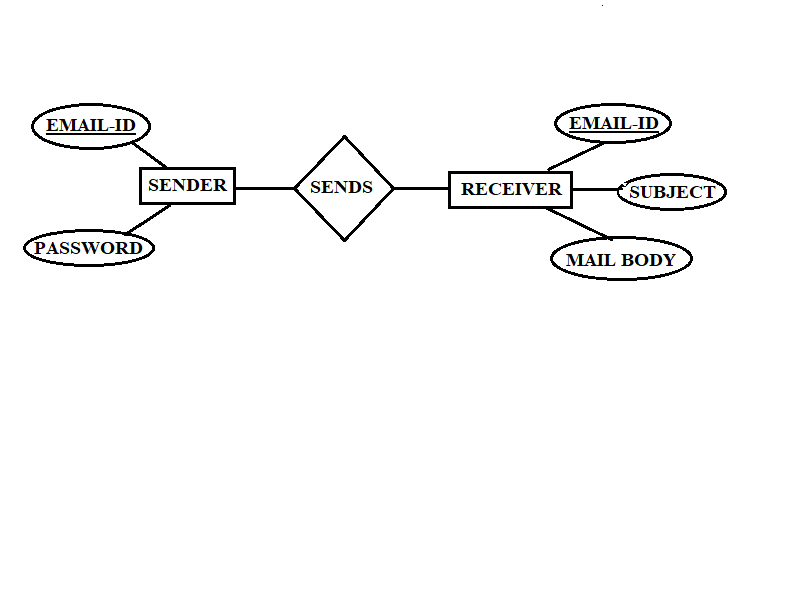
**MODULE DESCRIPTION**

* **smtplib:** The **smtplib** module defines an **SMTP** client session object that can be used to send mail to any internet machine with an **SMTP** or ESMTP listener daemon.
* **speech\_recognition:** Library for performing **speech recognition**, with support for several engines and APIs, online and offline.
* **pyttsx3:pyttsx3** is a text-to-speech conversion library in **Python**. Unlike alternative libraries, it works offline, and is compatible with both **Python** 2 and 3.
* **email.message:** It is **the base class for the email object model**. EmailMessage provides the core functionality for setting and querying header fields, for accessing message bodies, and for creating or modifying structured messages. An email message consists of headers and a payload (which is also referred to as the content).
* **tkinter:** Tkinter is **the de facto way in Python to create Graphical User interfaces (GUIs)** and is included in all standard Python distributions. In fact, it's the only framework built into the Python standard library.
* **mysql.connector:** MySQL Connector/Python enables Python programs to access MySQL databases, using an API that is compliant with the Python Database API Specification.
* **Datetime:** The datetime module supplies classes for manipulating dates and times. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. General calendar related functions.

**DATA FLOW DIAGRAM**



**ENTITY RELATIONSHIP DIAGRAM**



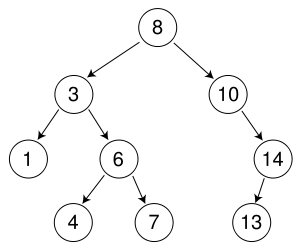
**DATABASE DESIGN**

A database is **information that is set up for easy access, management and updating**. Computer databases typically store aggregations of data records or files that contain information, such as sales transactions, customer data, financials and product information.

**DATABASE STRUCTURE:** TREE STRUCTURE

**INTRODUCTION:** Trees are a data structures that **link nodes in a parent/child relationship**, in the sense that there're nodes that depend on or come off other nodes. Trees are formed by a root node (the first node on the tree), and all the nodes that come off that root are called children

**PURPOSE:** A tree data structure is **an algorithm for placing and locating files (called records or keys) in a database**. The algorithm finds data by repeatedly making choices at decision points called nodes. A node can have as few as two branches (also called children) or as many as several dozen. Making it easier and more reliable for the user hence serving its actual purpose.



**DATABASE TABLE**

Tables are **database objects that contain all the data in a database**. In tables, data is logically organized in a row-and-column format similar to a spreadsheet. Each row represents a unique record, and each column represents a field in the record.

**DATABASE NAME:** JINGLE

**TABLE NAME:** TABLEX

**TABLE CONTENTS:-**

* **NAME:** Holds the name of recipient to whom mail is sent.
* **EMAIL\_ADDRESS:** Holds the mail id of the recipient**.**
* **SUBJECT:** Holds the subject of the mail.
* **MAILING\_DATE:** Date on which mail is sent.

**CODE**

#creating objects of diff modules

TTS=pyttsx3.init() #for text to speech conversion

STT=sr.Recognizer() #for speech recognition

win= Tk() #for GUI

#function where bot speaks given text

def botSpeaks(Audio):

    TTS.setProperty("rate",200)

    TTS.say(Audio)

    TTS.runAndWait()

#fucntion to recognize user's voice

def userSpeaks():

    try:

        with sr.Microphone() as source:

            print("Listening")

            s="Listening"

            STT.pause\_threshold=1

            STT.energy\_threshold=1500

            hear=STT.listen(source)

            info=STT.recognize\_google(hear)

            print(info)

            return info

    except Exception as e:

            print(e)

            print("Try Again...Not Recognisable")

            #m1=Label(win,text="Try Again...Not Recognisable",font='terminal',fg='gold',bg='black').grid(row=3,column=2,padx=10,pady=10)

            return "None"

#function to create and add data in database

def db\_get(name,reciever,subject,ms\_tm):

    nm=name

    eid=reciever

    sub=subject

    mess\_timing=ms\_tm

    obj=msl.connect(host='localhost',password='Khushi@2702',user='root',database='jingle')

    cursor=obj.cursor()

    cursor.execute('create table if not exists tableX(name varchar(30),e\_id varchar(30),subject varchar (100),time datetime)')

    cursor.execute('insert into tableX values(%s,%s,%s,%s)',(nm,eid,sub,mess\_timing))

    obj.commit()

    cursor.execute('select \* from tableX')

    print("inserted")

    for i in cursor:

        print(i)

#function that sends mail to recipient acc to info given

def sendMail(name,reciever,subject,message):

    server=smtplib.SMTP('smtp.gmail.com',587)

    server.starttls()

    myemail='spare.ac.2702@gmail.com'

    password='fqmyhmmqivaeepwn'

    server.login(user=myemail,password=password)

    mail=EmailMessage()

    mail['From']='spare.ac.2702@gmail.com'

    mail['To']= reciever

    mail['Subject']=subject

    mail.set\_content(message)

    server.send\_message(mail)

    now=datetime.now()

    ms\_tm=now.strftime('%y-%m-%d %H:%M:%S')

    print("Message sent at:",ms\_tm)

    m3=Label(win,text="Message Sent!",font='terminal',fg='gold',bg='black').grid(row=5,column=2,padx=10,pady=10)

    db\_get(name,reciever,subject,ms\_tm)

#dictionary acting as contact list

nicknames={'Khushi': 'gkhushiask6@gmail.com',

        'Shivangi': 'shivangiyadav20bca024@gmail.com',

            'Shristi': 'shrishtipal20bca052@gmail.com'

            }

#final bot method which ask for info from the user and send message

def getInfo():

    botSpeaks('Speak the name or email address whom you want to send email')

    name=userSpeaks()

    if name in nicknames:

        reciever=nicknames[name]

        print(reciever)

    else:

        reciever=input()

    botSpeaks("Speak the subject of your mail")

    subject=userSpeaks()

    botSpeaks('Speak the body of your mail')

    message=userSpeaks()

    sendMail(name,reciever,subject,message)

    botSpeaks("is there anything else I can do")

    reply=userSpeaks()

    if 'yes' in reply:

        getInfo()

    else:

        print('Thank You!')

        m2=Label(win,text="Thank You!",font='terminal',fg='gold',bg='black').grid(row=6,column=2,padx=10,pady=10)

#function creating database display window

def db():

    win=Tk()

    win.title('JINGLE DATA')

    win.geometry('1110x500')

    obj=msl.connect(host='localhost',password='Khushi@2702',user='root',database='jingle')

    print('connection established')

    cursor=obj.cursor()

    cursor.execute('SELECT \* from tableX')

    tree=ttk.Treeview(win)

    tree['show']='headings'

    tree['columns']=('1','2','3','4')

    tree.column('1',width=100,minwidth=50,anchor=CENTER)

    tree.column('2',width=300,minwidth=50,anchor=CENTER)

    tree.column('3',width=500,minwidth=50,anchor=CENTER)

    tree.column('4',width=200,minwidth=50,anchor=CENTER)

    tree.heading("1",text="NAME",anchor=CENTER)

    tree.heading("2",text="EMAIL ID",anchor=CENTER)

    tree.heading("3",text="SUBJECT",anchor=CENTER)

    tree.heading("4",text="DATE",anchor=CENTER)

    i=0

    for row in cursor:

        tree.insert('',i,values=(row[0],row[1],row[2],row[3]))

        i=i+1

    tree.place(x=0,y=0)

    win.mainloop()

#GUI

p=PhotoImage(file=r'C:\Users\Shridhar Gupta\Downloads\sym.png')

win.iconphoto(False,p)

win.title("JINGLE")

h=Label(win,text="Hi there! Click to begin :)",font='terminal',fg = 'gold', bg='black').grid(row=1,column=2,padx=10,pady=10)

b=Button(win,text='Click',font='terminal',fg='black',bg='gold',command=getInfo).grid(row=2,column=2,padx=10,pady=10)

pb=Button(win,text='Preview',font='terminal',fg='black',bg='gold',command=db).grid(row=3,column=2,padx=10,pady=10)

win.configure(background='black')

win.geometry('350x320+900+350')

win.mainloop()

**TESTING**

Software testing is **the process of evaluating and verifying that a software product or application does what it is supposed to do**. The benefits of testing include preventing bugs, reducing development costs and improving performance.

**OBJECTIVE OF TESTING:**  Finding defects which may get created by the programmer while developing the software. Gaining confidence in and providing information about the level of quality. To prevent defects.

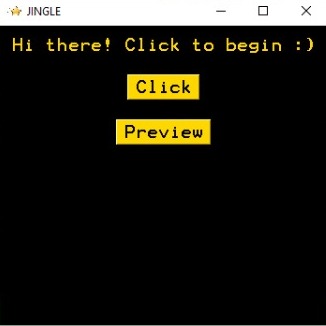
**TYPES OF TESTING IMPLEMENTED:-**

* **UNIT TESTING:** Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. This testing methodology is done during the development process by the software developers and sometimes QA staff.
* **INTEGRATION TESTING:** Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements.
* **SYSTEM TESTING:** System testing, also referred to as system-level tests or system-integration testing, is the process in which a quality

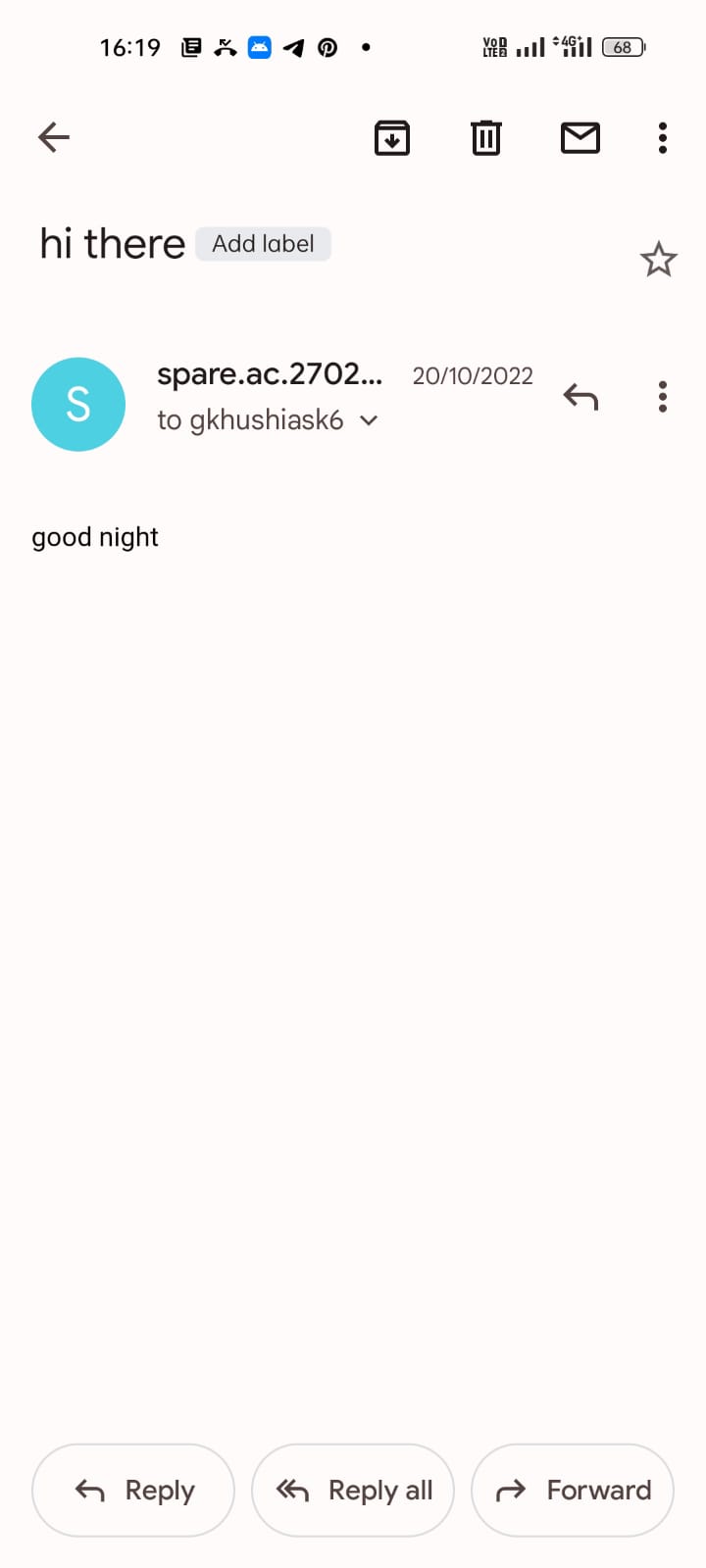
assurance (QA) team evaluates how the various components of an application interact together in the full, integrated system or application.

* **USER ACCEPTANCE TESTING:** User acceptance testing (UAT), also called application testing or end-user testing, is a phase of software development in which the software is tested in the real world by its intended audience.
* **VALIDATION:** It is the static practice of studying and verifying the specific requirements of a particular stage in development. It is the dynamic practice of testing the final product after development to check that it meets customer requirements. It does not require executing code.
* **OUTPUT TESTING:** It is a testing technique where combinations of input values are tested in a systematic way. Pair-wise Testing - The behaviour of software depends on multiple parameters. In pairwise testing, the multiple parameters are tested pair-wise for their different values.

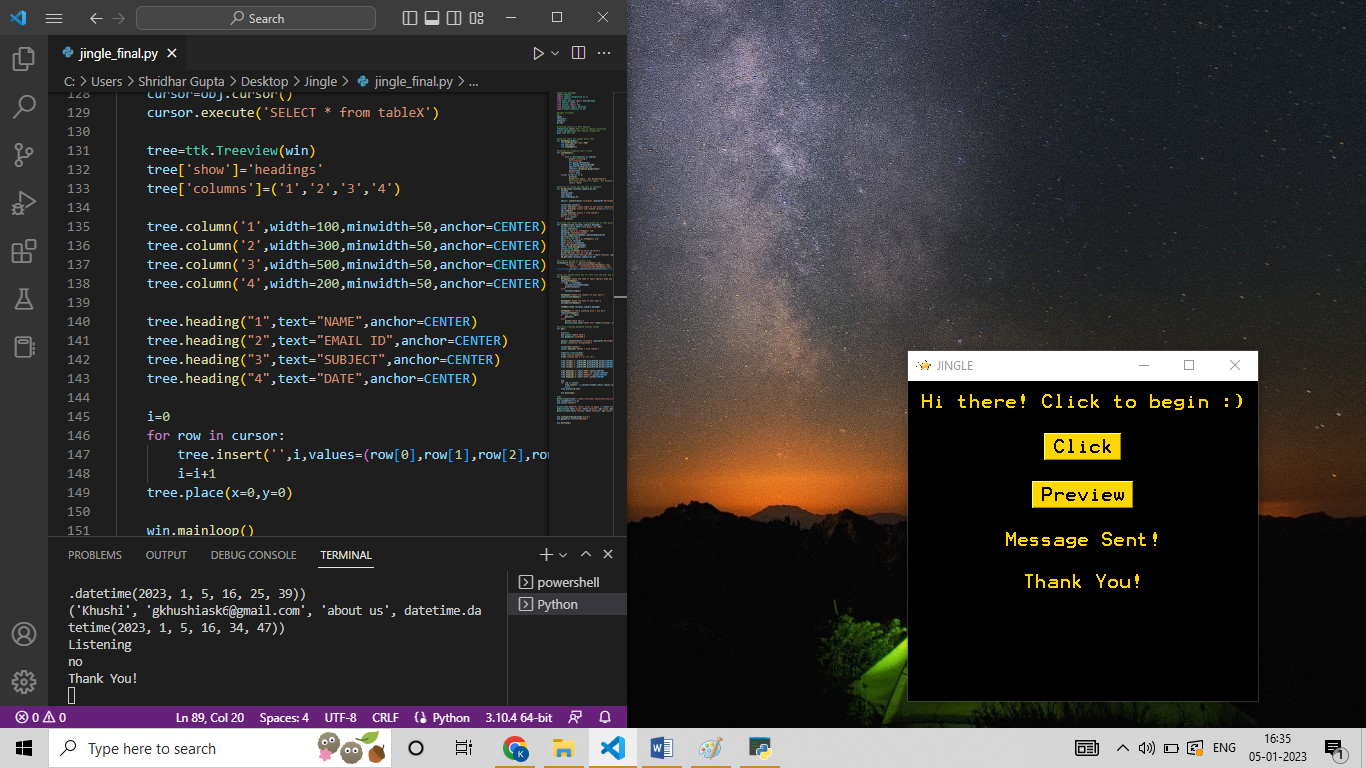
**SCREENSHOTS**

****

**OPENING WINDOW**

****

**SENT MESSAGE**



**WINDOW AFTER MESSAGE IS SENT**

****

**MAIL LOG (DATABSE)**

**FUTURE SCOPE OF THE PROJECT**

Technology has changed our day-to-day lives. Technology has brought the world closer and better connected. Those days have passed when only the rich could afford such luxuries. Because of the rise of globalisation and liberalisation, all luxuries are now within the reach of the average person. Today, an average middle-class family can afford a mobile phone, a television, a washing machine, a refrigerator, a computer, the Internet, etc. At the touch of a switch, a man can witness any event that is happening in far-off places.

Technology has reduced the effort and time and increased the efficiency of the production requirements in every field. Jingle is one such technological approach which has made our lives easy, comfortable, and enjoyable. It has brought a revolution in and communication. With the innovation of a particular technology, it becomes part of society and integral to human lives after a point in time.

**With Jingle, in coming future, we are planning to bring various modifications in its technology as well as the gadgets in which it is used i.e. at present jingle works only in PC’s but the plan is to make it platform independent , we’ll be able to see it running in our smartphones, smartwatches and even personalized robots. We have also planned to bring other features such as better AI, time allotted mailing and much more. Since jingle at the moment is only focused in mailing services with the passing times we’re planning to expand its use in probably every messaging platform.**

**CONCLUSION**

**JINGLE** is a GUI enhanced chat bot designed using Python in front end and MySQL in the backend. Python libraries are responsible for graphical user interface, voice recognition and mailing facilities using SMTP sever. Whereas MySQL is responsible for creating and maintaining the database holding the mailing data.

Jingle mainly focuses to provide a user friendly environment to its users, to ease out the day to day responsibilities for its user by keeping a track of mailing records which can easily be handled using voice commands.

**BIBLIOGRAPHY**

References of this project have been taken from different Youtube channels like Amit Thinks, Programming Hero and so on for amendment of minor errors and acquiring better knowledge on some methodologies and also while carrying out testing processes.

Google has also been referenced for GUI resources, as images from sites like pngtree.com and pixabay.com and Pinterest has been used for referencing basic colour palettes for the background of the application window and the fonts used in the window to make it more user likable. Sites like Stackoverflow.com, w3Schools.com, TutorialsPoint.com etc. have come in handy to get a thorough knowledge of the modules used both in Python and MySQL which allowed us to add more creativity and wider brackets in the application.