UNIVERSITY OF MUMBAI

A Project Report On

VIRTUAL ASSISTANT

Submitted by

Mr. Bhavik Rajendra Sutar

In partial fulfillment of getting degree of B.sc. in Information Technology.

Under The Guidance Of

Dr. Ajay Barve

Kirti M Doongursee College of Arts, Commerce and Science

Academic Year: 2021-2022

KIRTI M. DOONGURSEE COLLEGE, DADAR (W), MUMBAI -400028.

CERTIFICATE

This is to Certify that, Mr. BhaviK Sutar

With Seat No.88

Has Successfully completed project entitled

VIRTUAL ASSISTANT

In the partial fulfillment of B.Sc. (Information Technology) Degree (Semester VI) has completed with all required Phases of the project development satisfactorily in the academic year 2021 – 2022

Project Guide **Head of the Department**

(Dr. Ajay Barve)

External Examiner

Date:

Date: **Date:**

INDEX

➢ OBJECTIVE & SCOPE OF THE PROJECTS

○ OBJECTIVE

❖ The main purpose of an intelligent virtual assistant is to answer questions that users may have in their daily life.

❖ One of the main advantages of voice searches is their rapidity. In fact, voice is reputed to be four times faster than a written search. Due to which users can save their valuable time.

❖ Another objective of virtual assistant means that it will be open from the registered face, due to which no other person can use your personal or virtual assistant.

❖ Currently, the project aims to provide the Linux Users with a Virtual Assistant that would not only aid in their daily routine tasks like searching the web, extracting weather data, vocabulary help and many others but also help in automation of various activities.

❖ In the long run, we aim to develop a complete server assistant, by automating the entire server management process - deployment, backups, auto-scaling, logging, monitoring and make it smart enough to act as a replacement for a 6 general server administrator.

- **SCOPE**

❖The virtual assistant will work by taking commands from the user and will give interesting information related to the search.

❖ Virtual assistant is able to open many desktop applications on the users desktop and able to perform many google search activities.

❖ The virtual assistant performs many operations such as task execution and web scraping and much more,which saves users time and helps a lot in usage.

❖ Virtual assistants will continue to offer more individual experience to the user.

❖ The virtual assistant is a remote bot that is trained and experienced to help users with the variety of the task.

❖ The virtual assistant is an artificial intelligence based smart bot and offers voice command and audio recognition to the user. the user is able to perform many smart activities with the help of virtual assistants.

➢ DEFINITION

❖We already have multiple virtual assistants. Like google assistant and Siri but they are on specific operating systems. Like google assistant works on ‘Android’ devices better than desktop systems and the Siri works only on ‘IOS’ based devices.

❖There are number of people who have issues in voice recognition. These systems can understand English phrases but they fail to recognize our accent. Our way of pronunciation is way distinct from theirs.

❖They are easier to use on mobile devices than desktop systems. There is a need of a virtual assistant that can understand English in an Indian accent and work properly on the desktop system.

❖When the virtual assistant is not able to answer questions accurately, it’s because it lacks the proper context or does not understand the intent of the question.

➢ EXISTING SYSTEM

The term virtual assistant, or virtual personal assistant, is also commonly used to describe contract workers who work from home doing administrative tasks typically performed by executive assistants or secretaries. Virtual assistants are typically cloud-based programs that require internet-connected devices and/or applications to work.

Three such applications are Siri on Apple devices, Cortana on Microsoft Devices and Google Assistant on Android devices. There are also devices dedicated to providing virtual assistance. The most popular ones are available from Amazon, Google and Microsoft.

To use the Amazon Echo virtual assistant, called Alexa, users call out the wake word, "Alexa." A light on the device signals to the user it is ready to receive a command, which typically involves simple language requests, such as "what is the weather today," or "play pop music." Those requests are processed and stored in Amazon's cloud.

The technologies that power virtual assistants require massive amounts of data, which feeds artificial intelligence (AI) platforms, including machine learning, natural language processing and speech recognition platforms.

As the end user interacts with a virtual assistant, the AI programming uses sophisticated algorithms to learn from data input and become better at predicting the end user's needs.

➢ PROPOSED SYSTEM

To design a device that acts as a digital organizer to provide variety of services to its master. It will look at examples of intelligent programs with natural language processing that are currently available, with different categories of support, and examine the potential usefulness of one specific piece of software as a VPA.

It continues to expand its digital abilities in organizing events, ordering food, playing music, guiding services for travelling, game prediction etc. It is suggested that new technologies may soon make the idea of virtual personal assistants a reality.

Experiments conducted on this system, combined with user testing, have provided evidence that a basic program with machine learning algorithms in the form of a digital personal assistant. Using machine learning algorithms to iteratively learn user's preference for each theme based on quality feedback given by the user.

The concept of a virtual assistant which is a digital service looking after a range of our needs is fast becoming a reality. As artificial intelligence and machine learning progress at pace, digital assistants are set to become our gateway to the internet and know more about us than we do ourselves.

Siri and Google now are just the beginning. The device accepts voice input processes it through various machine learning algorithms to provide desired output to user.

➢ STATEMENT OF NEED

❖Virtual assistant software is required to act as an interface into the digital world by understanding user requests or commands and then translating into actions or recommendations based on agent's understanding of the world 12

❖The virtual assistant enables hands-free use of many applications. For this the virtual assistant is provided with the path of the applications to contact them.

❖Virtual assistants provide a wide range of services. But for this it is necessary that the programs given in it should work properly.

❖Sometimes the virtual assistant can get stuck in listening mode, for it to have a good connection to the system's microphone and it must also have the threshold value set for background noise cancellation.

❖Features like Face Verification and Speak require a good connection of the virtual assistant with the system's camera and speaker and most importantly, it must have a good connection to the Internet

➢ HARDWARE & SOFTWARE USED

○ HARDWARE

❖Minimum 2 GB of ram

❖Sound card, Speakers

❖Microphone

❖1Ghz-2Ghz processor

○ SOFTWARE

The software requirement for the virtual assistant is to be light-weight So do not be a burden on the processor of the machine. All software requirements are here:

❖Windows 7 or above

❖VS Code editor

❖Python 3

❖Excel (csv file support)

➢ EXPLANITION OF TECHNOLOGY

❖Virtual assistant is going to be a GUI (Graphical User Interface) Application and it will provide the user very mannered and interactive responses.

❖It is capable of voice interaction and it is able to do many activities such as Music playback, YouTube search and many more activities.

❖Due to it offers voice search this will help the user to simplify many things.

❖Virtual assistant is a personal desktop bot which is capable of voice searching and task execution. It is an AI (Artificial intelligence) based bot and the virtual assistant is capable of voice interaction with the user. 7

❖It is able to take notes by voice command and is also able to send the mail in a mannered way which is the receiver, subject and message of the mail.

❖In today's world, everyone keeps on trying to make their day-to-day work easier. The virtual assistant helps to the user does the same job as it is related to internet and IOT based products

➢ SYSTEM DEVLOPMENT LIFE CYCLE

The main goal of the CryptSDLC architecture is to support the development process of cryptographic applications. It is a conceptual approach to structure and streamline typical tasks identified in cryptographic engineering. The model helps to cope with the complexity and interdisciplinary nature of cryptographic application design. It is based on the experiences made in an EU research project with people from very different disciplines involved, all targeted at a single goal: the development of cloud services which are secure by design and leverage feasible cryptography. i. Primitives Layers. The lowest layer consists of cryptographic primitives and protocols which represent basic cryptographic building blocks, e.g., signature schemes, or cryptographic protocols. ii. Tools Layer. The second layer is denoted as tools layer. Tools are a concept introduced by CryptSDLC to communicate techniques to software developers and architects in an more accessible way. They provide higher level functionality as a combination of primitives which serve a particular purpose and also come with an accompanying implementation in form of, e.g., software libraries. iii. Service Layer. Cloud computing is radically changing the way we are consuming IT resources but also influencing the way software is built and deployedModern applications try to leverage the idea of service oriented architectures to support scalability and elasticity through modularization. This gives the required freedom in deployment needed in modern cloud environment, be it private, public or hybrid cloud settings.

➢ FEASIBILITY STUDY

Feasibility study analytics that determine whether all relevant factors are taken into account. Including Technical feasibility, Operational feasibility and Economical feasibility. 13 Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration. The facts taken into consideration in the Facility Study are:

❖**Technical feasibility**: It includes finding out technologies for the project, both hardware and software. For virtual assistant, the system must have a microphone to convey the user's message and a speaker to listen when the system speaks. These are very cheap nowadays and every system comes with them also everyone generally possess them. Besides, the system needs an internet connection.While using virtual assistant, make sure the system has a steady internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.

❖**Operational feasibility**: It is the ease and simplicity of operation of the proposed system.System does not require any special 14 skill set for users to operate it. In fact, it is designed to be used by almost everyone. People who don't like to read out the text virtual assistant will help them.

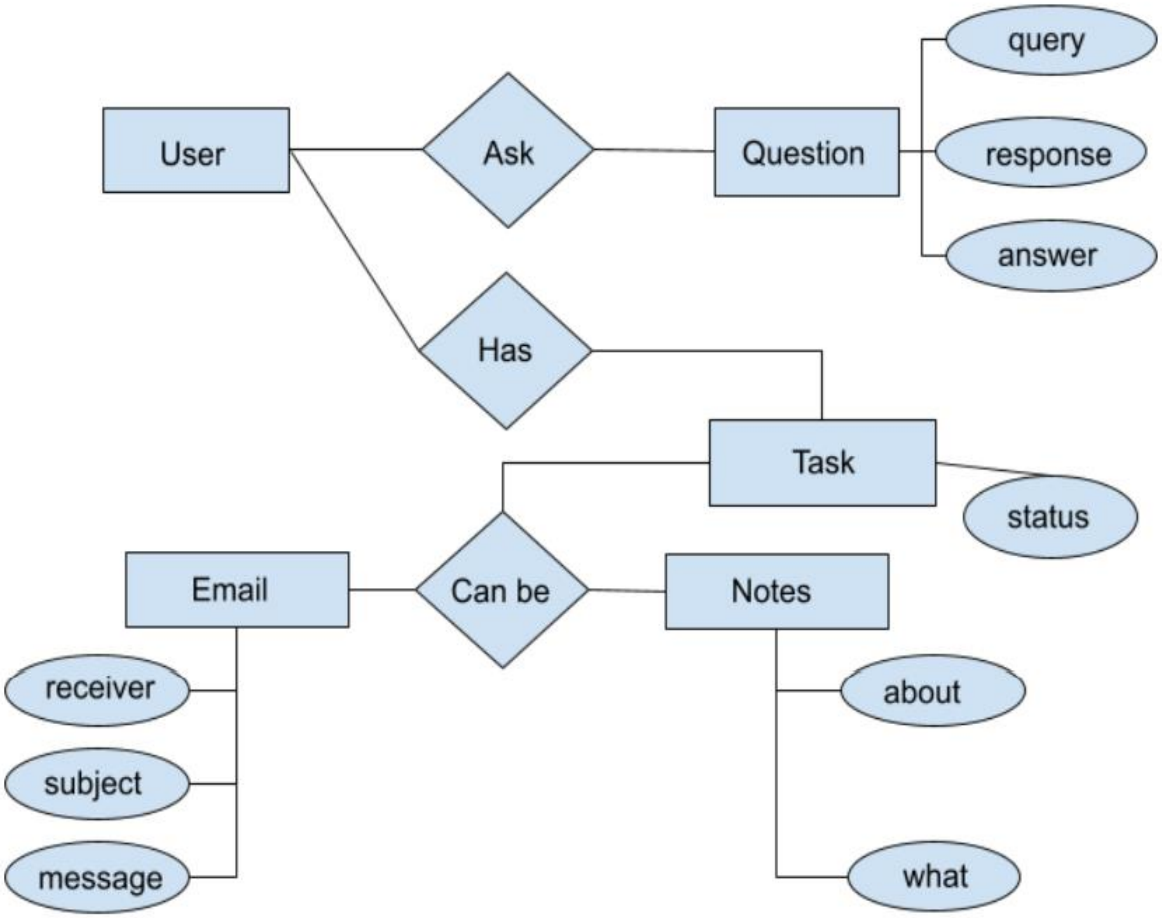
❖**Economical feasibility**: Economic feasibility is a kind of cost-benefit analysis of the examined project, which assesses whether it is possible to implement it. Here,the total cost and benefit of the proposed system over the current system.For the project virtual assistant the user would have to pay for microphone, speaker and the camera nowadays many systems come along with an inbuilt system of the microphone, camera and speaker. Again, they are cheap and easily available in the market

➢ GANTT CHART

GANTT CHART :-

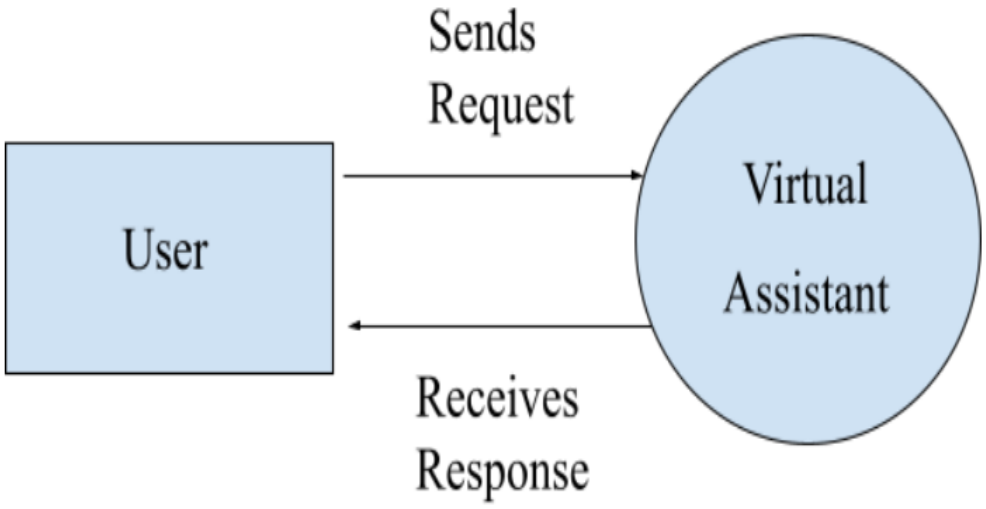
Activity	Time Frame					
	10/09/2021 To 26/09/2021	5/10/2021 To 20/10/2021	21/10/2021 To 20/11/2021	25/11/2021 To 28/12/2021	10/01/2022 To 12/02/2022	20/02/2022 To 25/03/2022
Literature Survey & Planning						
Feature Finalisation With Mockup						
Front End Developments						
BackEnd Development						
Testing & Fault Detection						
Project Report						

➢ ENTITY RELATIONSHIP DIAGRAM

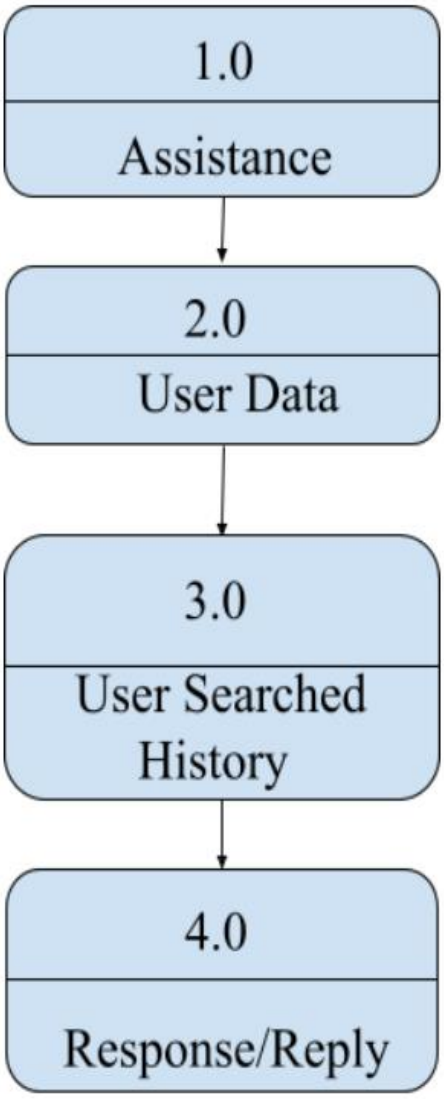


➢ DATA FLOW DIAGRAM

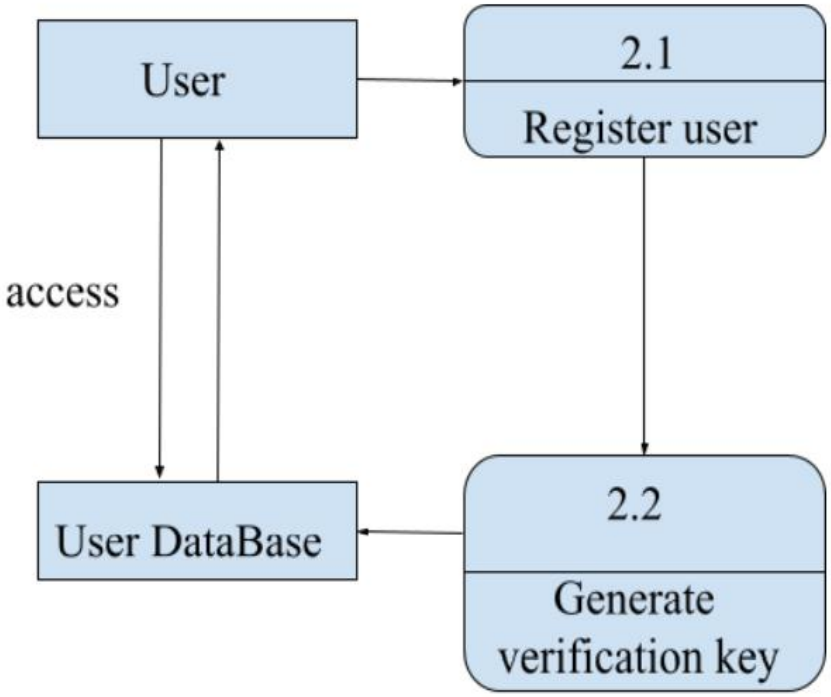
DFD Level 0 or Context level Diagram



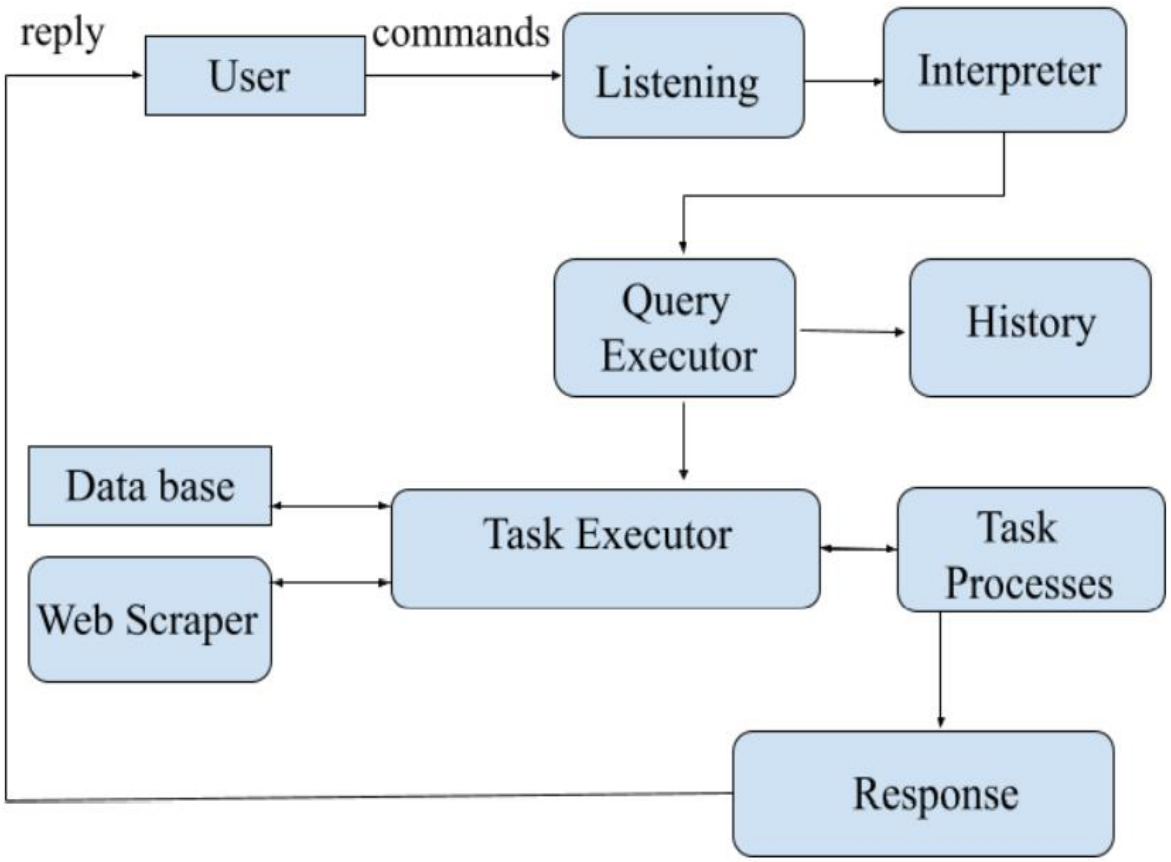
DFD Level 1


1.0
Assistance
2.0
User Data
3.0
User Searched History
4.0
Response/Reply

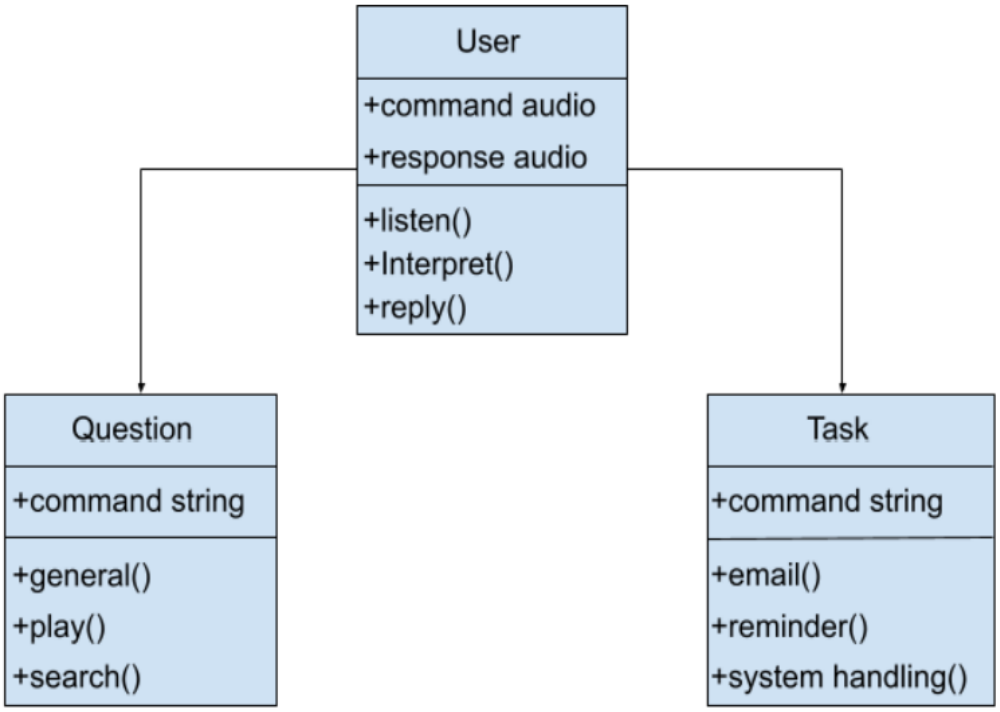
DFD Level 2


User
2.1
Register user
access
User DataBase
2.2
Generate
verification key

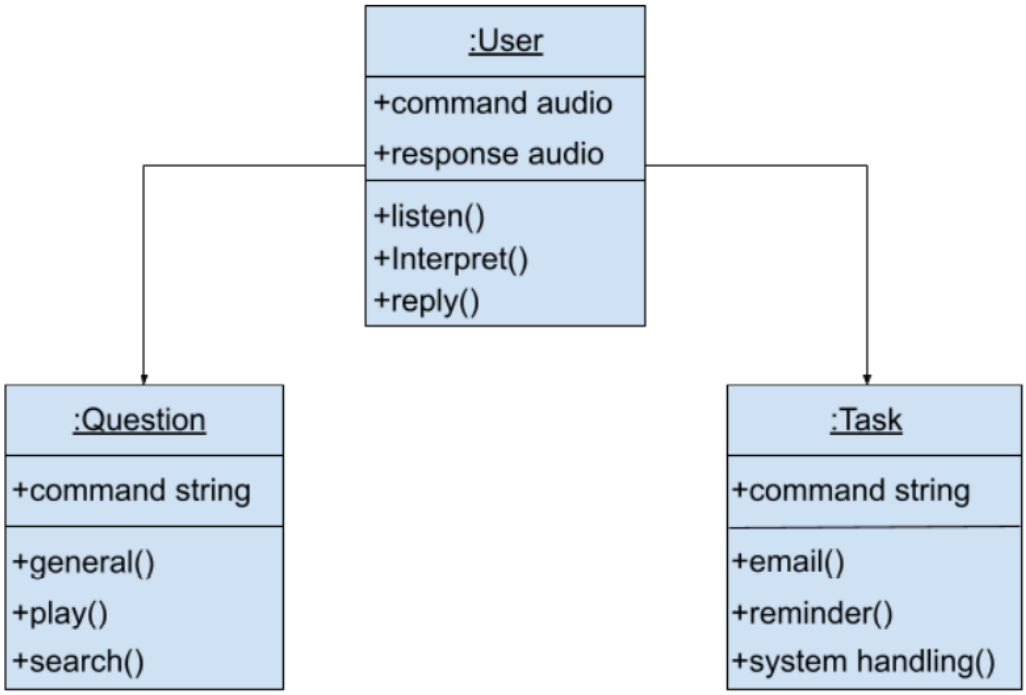
DFD Level 4


reply
User
commands
Listening
Interpreter
Query Executor
History
Data base
Web Scraper
Task Executor
Task Processes
Response

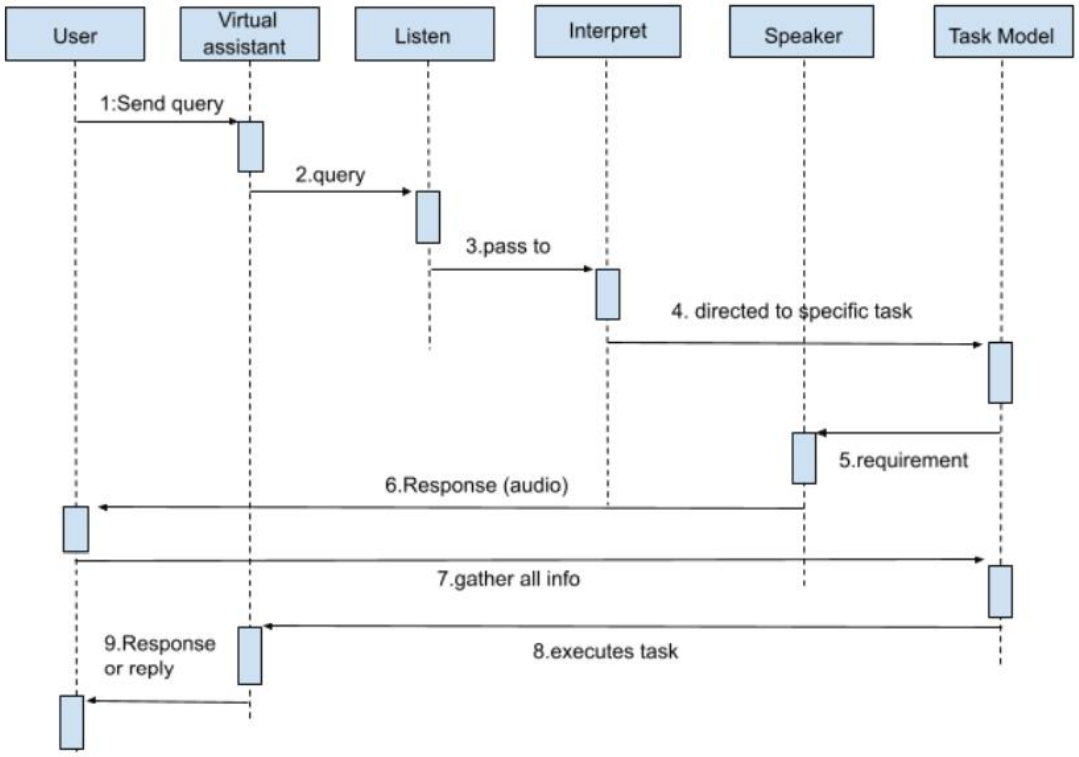
CLASS DIAGRAM



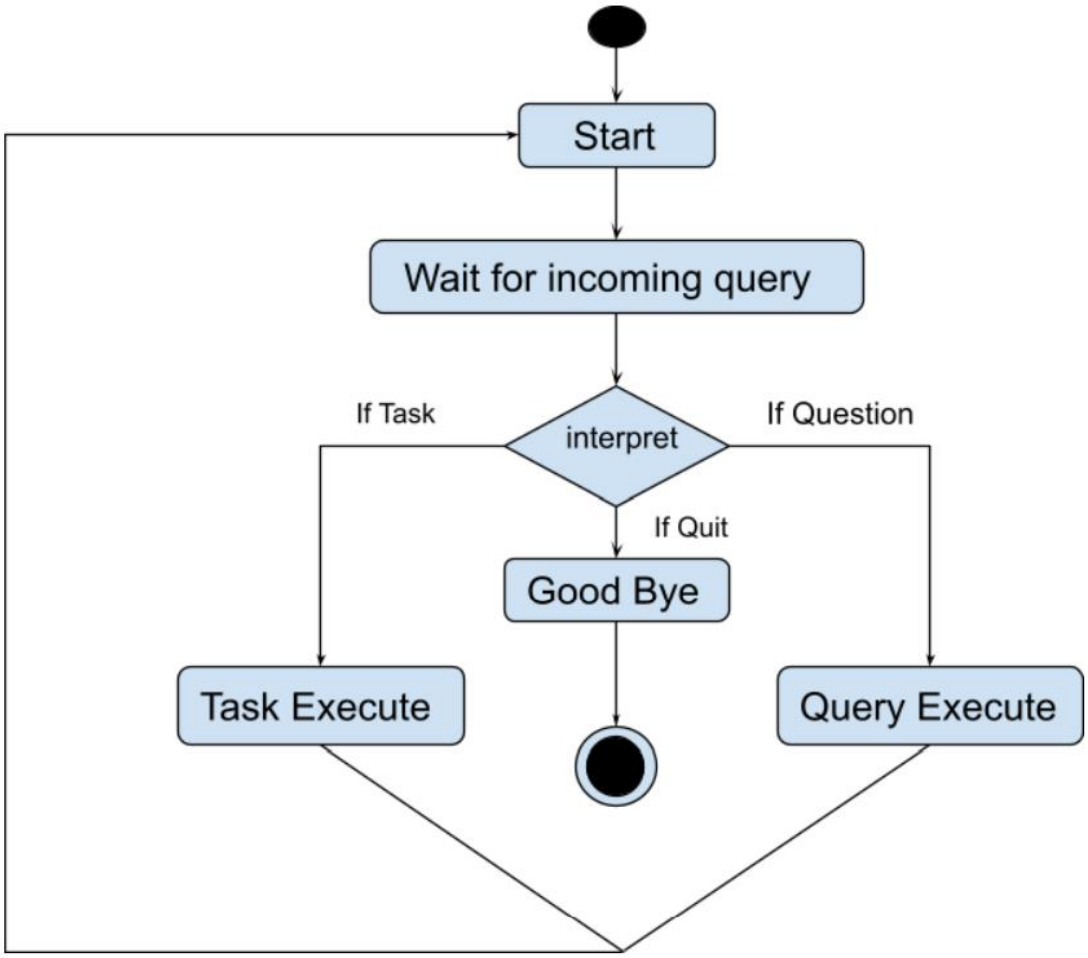
OBJECT DIAGRAM



SEQUENCE DIAGRAM

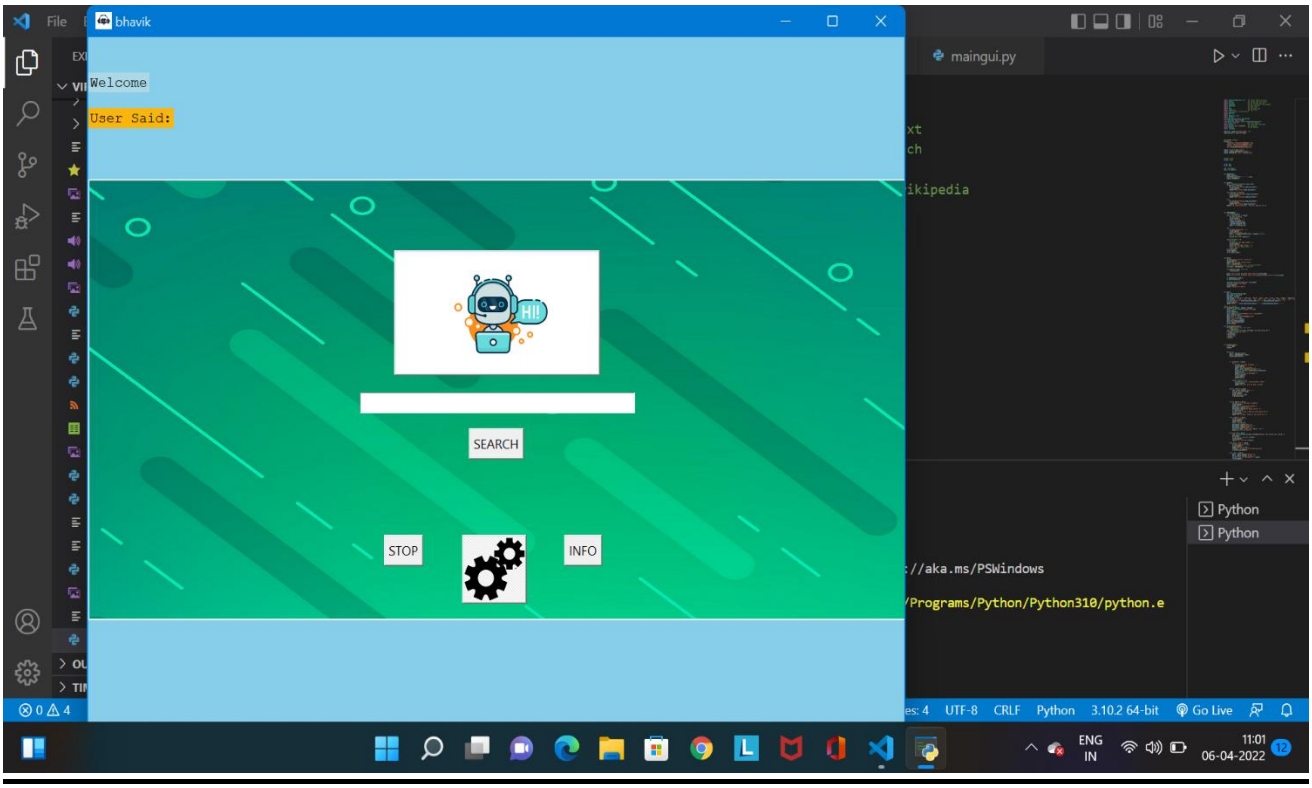


STATE CHART DIAGRAM



➢ SNAPSHOTS

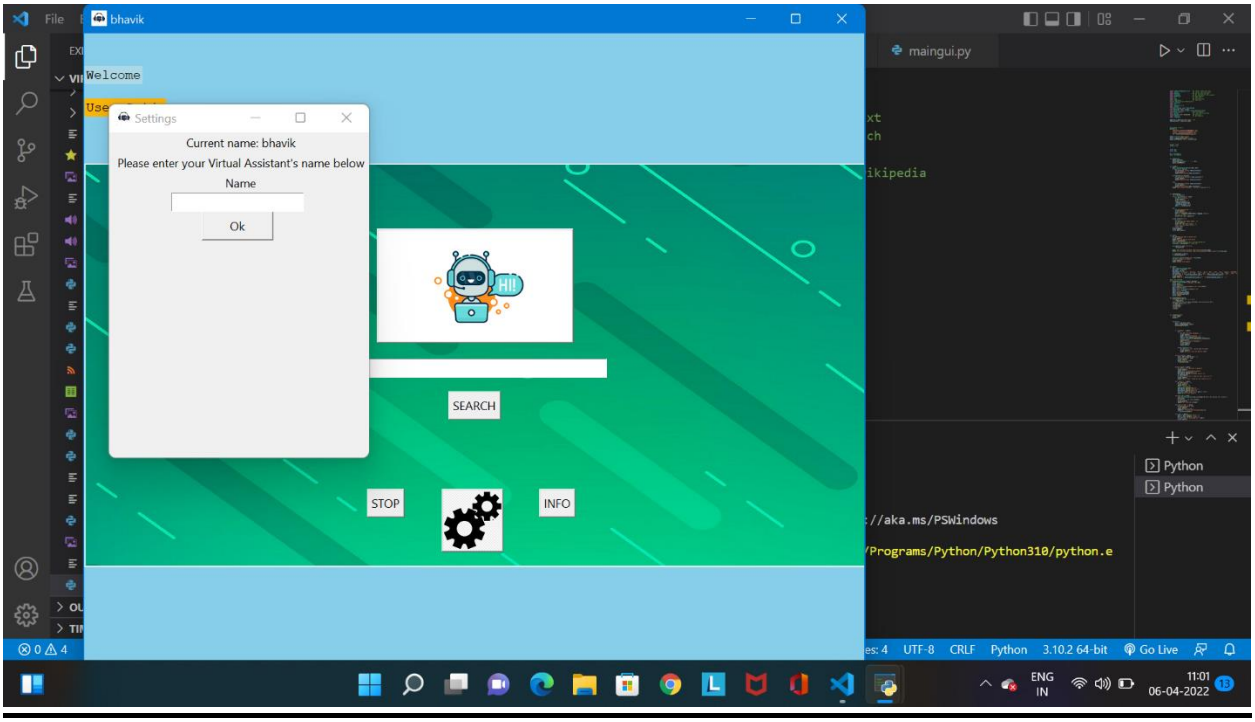
USER INTERFACE



The graphical user interface of the virtual assistant contains various buttons which are setting button, info button, stop button, type button, search button and to activate voice search assistant contains image button of bot. At the top side of the graphical user interface the virtual assistant contains two labels which are ‘Welcome’ and ‘User said’. In the user said input given by the user to the assistant will be shown and in the place of the welcome response from the assistant will be shown.

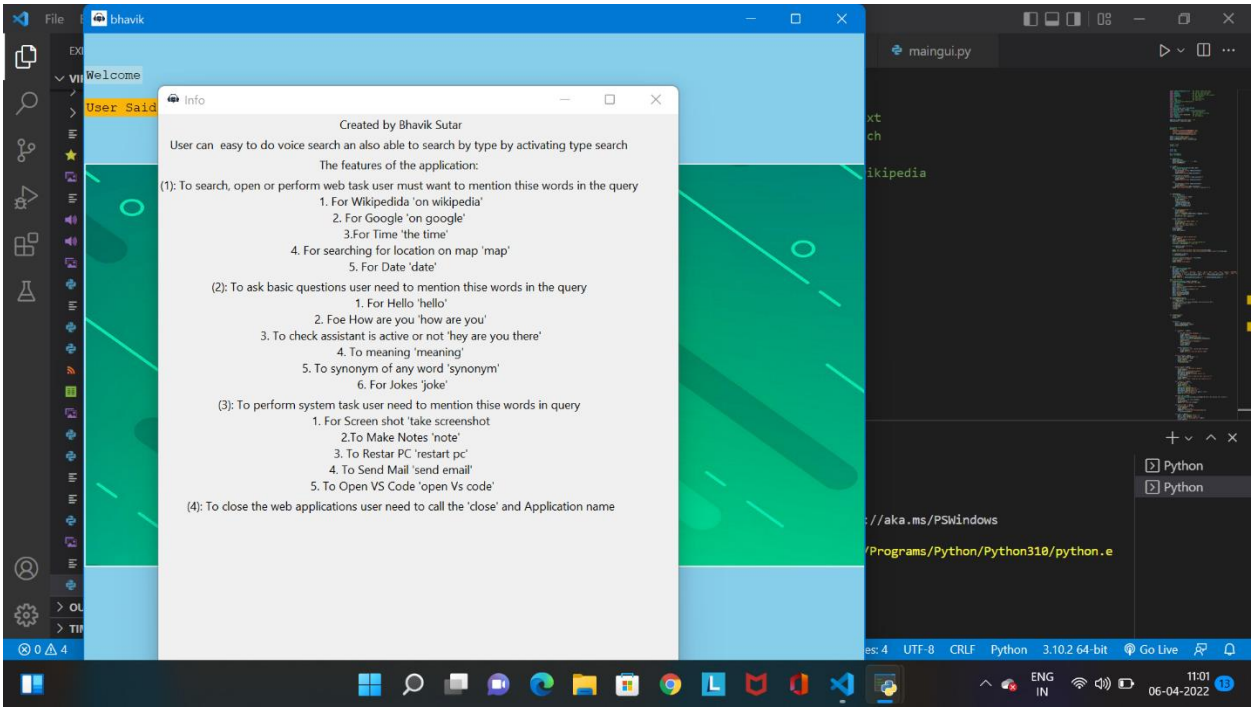
STOP BUTTON : Stop button is use to terminate the graphical user interface and the program.

SETTING WINDOW



SETTING BUTTON : With the help of the setting button, the user can give his name to the graphical user interface, for that he has to click on setting, then after that a setting window will open in that it will contain the current name of the GUI and a text field will be given and the ok button will be given below it the user just has to put his name in the text field and click on ok, then the program will terminate and on restarting the GUI user's name will be updated there.

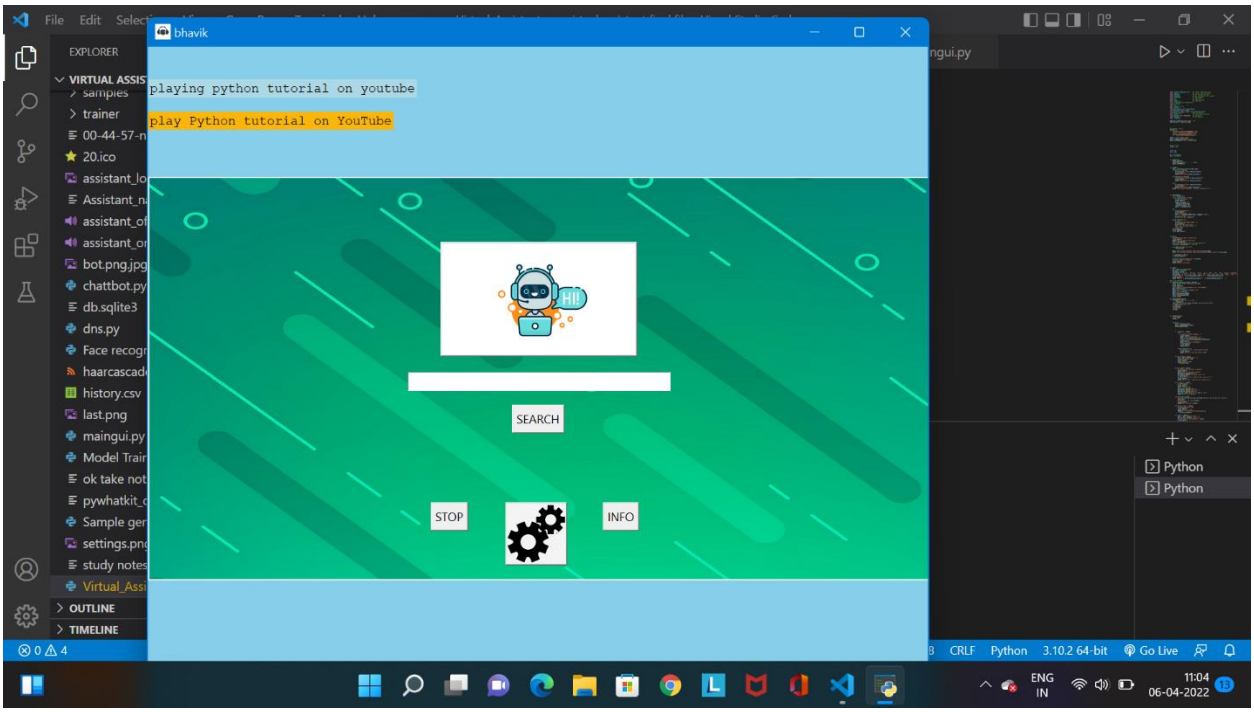
INFO WINDOW

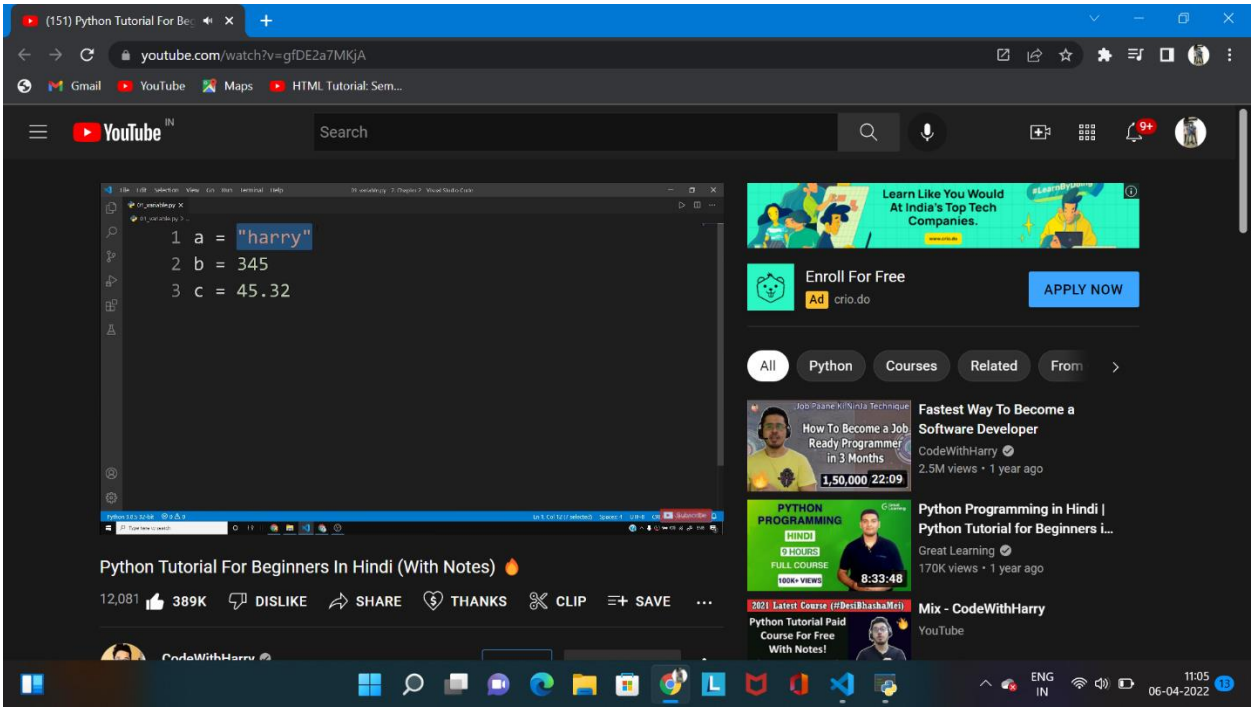


INFO BUTTON : The main aim of the info button is to help the user. After clicking on the info button info window will open and it will contains all the instruction are given on how to command the assistant

Youtube search result :

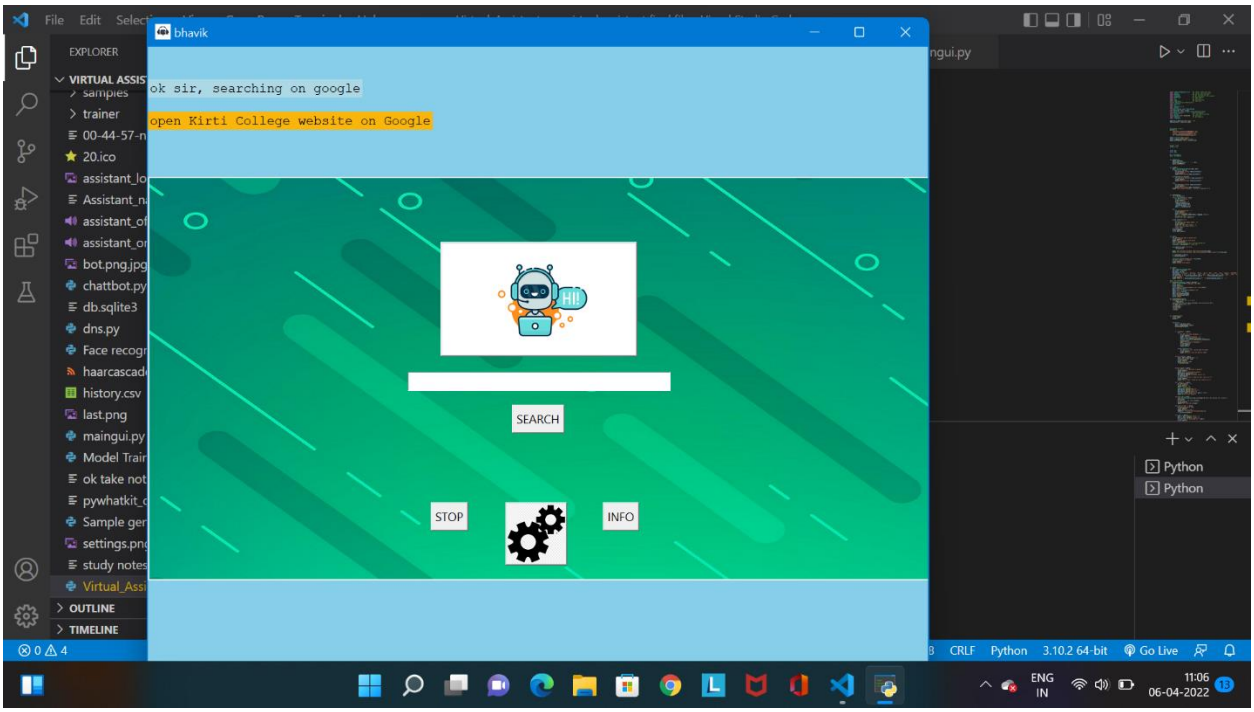
Searched query and it’s result on youtube

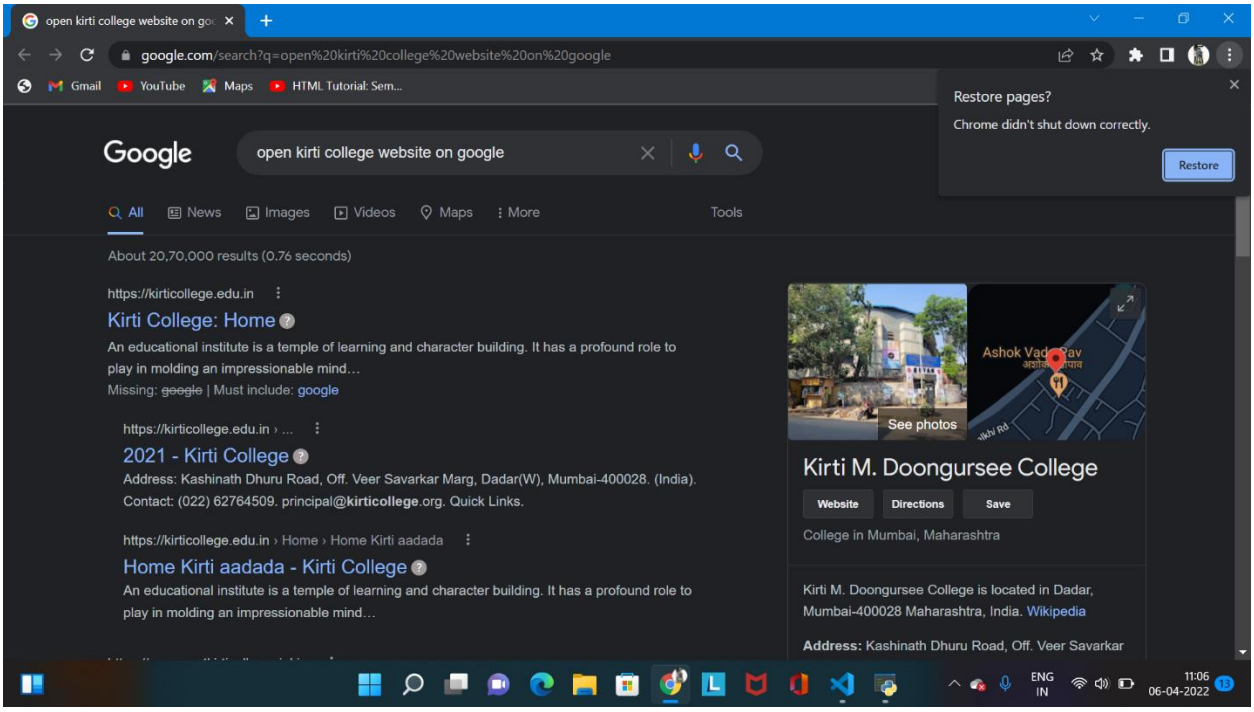




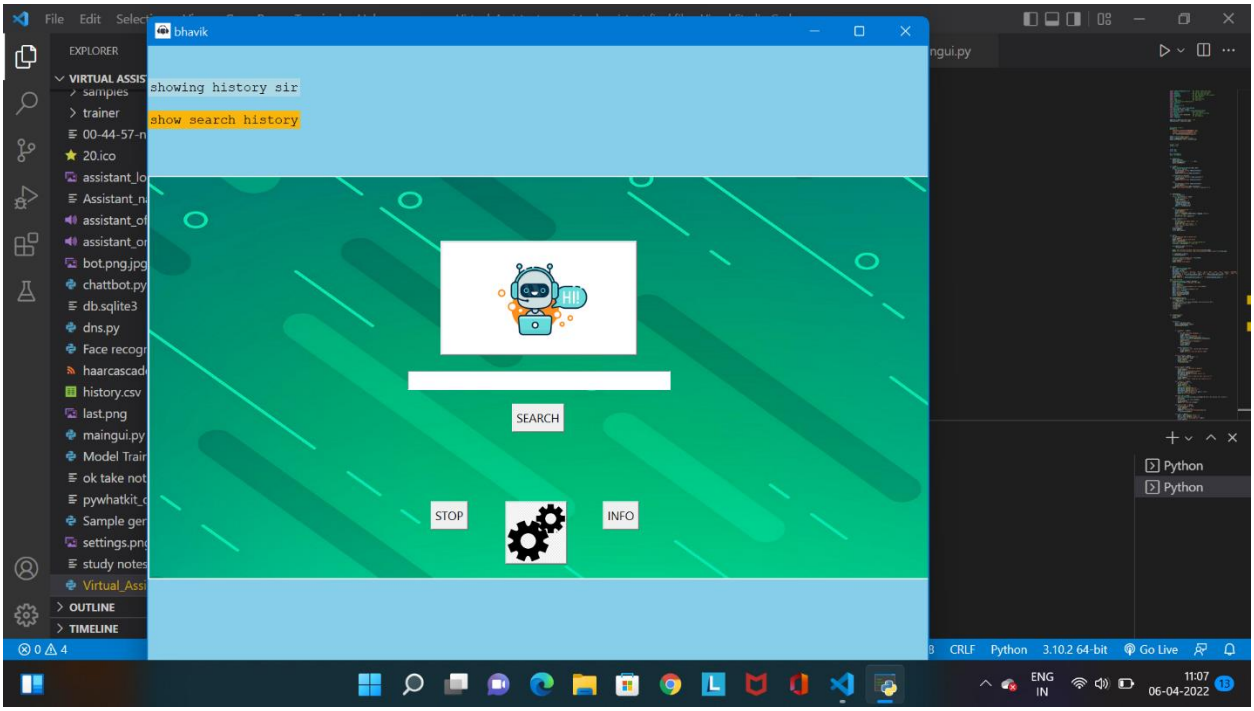
Google search result :

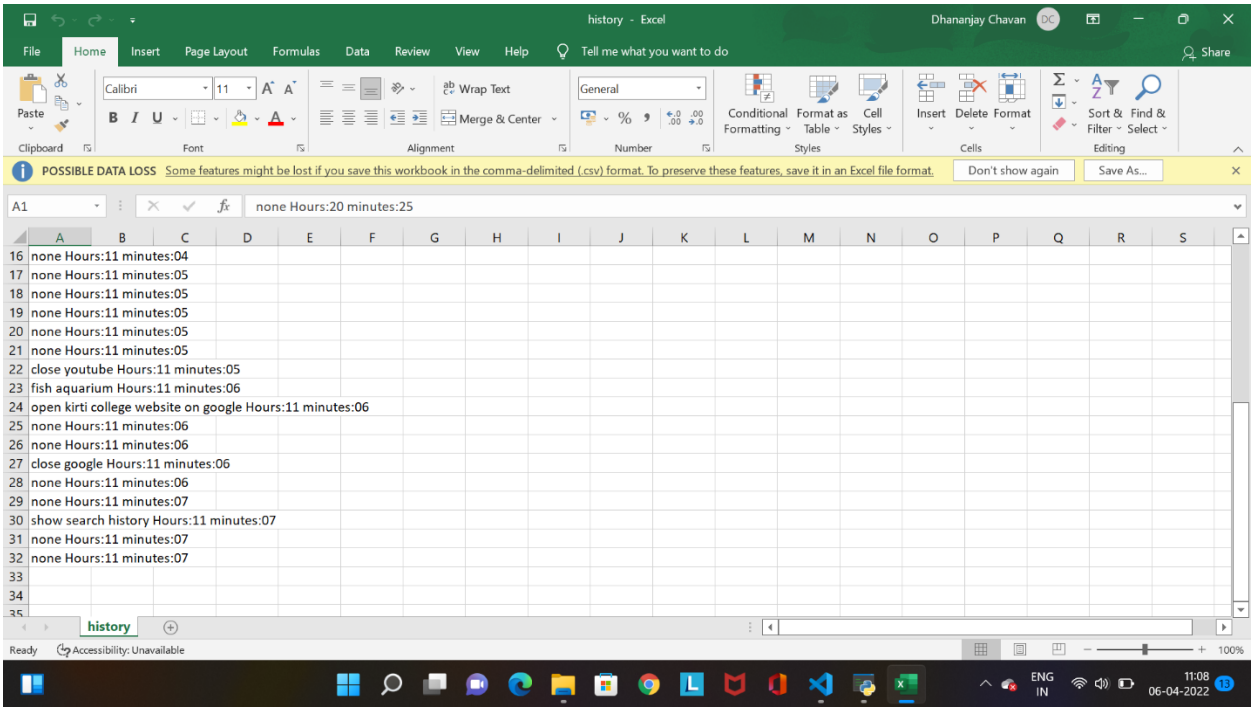
Searched query and it’s result on google





User search history





	A
16	none Hours:11 minutes:04
17	none Hours:11 minutes:05
18	none Hours:11 minutes:05
19	none Hours:11 minutes:05
20	none Hours:11 minutes:05
21	none Hours:11 minutes:05
22	close youtube Hours:11 minutes:05
23	fish aquarium Hours:11 minutes:06
24	open kirti college website on google Hours:11 minutes:06
25	none Hours:11 minutes:06
26	none Hours:11 minutes:06
27	close google Hours:11 minutes:06
28	none Hours:11 minutes:06
29	none Hours:11 minutes:07
30	show search history Hours:11 minutes:07
31	none Hours:11 minutes:07
32	none Hours:11 minutes:07
33	
34	

TESTING & IMPLEMENTATION

In the software testing process for testing virtual assistants quality control. The main goal of converter testing is to identify and discover defects and bugs in the virtual assistant and improve the stability and performance. The different types of testing are : 1) Performance testing. 2) System testing. 3) Unit testing

1)Performance testing : The overall performance of the virtual assistant system has been checked. The threading package of python is used to optimize the performance of the graphical user interface of the virtual assistant The important parameters check while performance testing :

- Network connectivity : The response time is checked over the various network speeds where it supports the network speed over (3G, 4G) and wifi connectivity.

- Response time of the virtual assistant is dependent on the network connectivity speed at a good speed it supports well. 53 2) System testing : The virtual assistant system is developed only for the Windows operating system. Here it works well on this operating system. 3) Unit testing : Every function of the program works well in the virtual assistant. The loop method of the function also works fine. If the system is unable to recognize or unable to take the user input it will notify the user.

SYSTEM CODING

```
import speech_recognition as sr    #To convert speech into text
import pyttsx3                     #To convert text into speech
import datetime                    #To get the date and time
import wikipedia                   #To get information from wikipedia
import webbrowser                  #To open websites
import os                          #To open files
import time                        #To calculate time
import subprocess                  #To open files
#import PyDictionary as PyDictionary
import pyautogui
import sys
import pywhatkit as kt
import smtplib
from email.message import EmailMessage
from chatterbot import ChatBot
from chatterbot.trainers import ChatterBotCorpusTrainer
from tkinter import *              #For the graphics
import pyjokes                     #For some really bad jokes
from playsound import playsound    #To playsound
import keyboard                    #To get keyboard
import threading


name_file = open("Assistant_name", "r")
name_assistant = name_file.read()
```

```
#DICTIONARY OF MAILS
mailDict ={
    "dhananjay":"dhananjaych0806@gmail.com",
    "bhavik":"bhaviksutar352002@gmail.com",
    "Chinya":"chinmaychavan07@gmail.com",
    "DC":"2019016401596644@mdcollege.in"}

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



#global screen
screen = Tk()


global var
global var1

var = StringVar()
var1 = StringVar()


def speak(text):
    engine.say(text)
    print(name_assistant + " : "  +  text)
```

```
    engine.runAndWait()



def wishMe():
    hour = int(datetime.datetime.now().hour)
    if hour>=0 and hour<12:
        var.set(f"Good morning {name_assistant}")
        screen.update()
        speak(f"Good morning {name_assistant}")

    elif hour>=12 and hour<18:
        var.set(f"Good afternoon {name_assistant}")
        screen.update()
        speak(f"Good afternoon {name_assistant}")

    else:
        var.set(f"Good evening {name_assistant}")
        screen.update()
        speak(f"Good evening {name_assistant}")
    speak("i am virtual assistant,   how can i help you sir!")




def takeCommand():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        var.set("Listening....")
```

```
        screen.update()
        print("Listening....")
        r.pause_threshold = 1
        r.energy_threshold=4000
        r.phrase_threshold = 0.8
        audio = r.listen(source)

    try:
        var.set("Recognizing....")
        screen.update()
        print("Recognizing....")
        query = r.recognize_google(audio, language ='en-in')
        #if 'me' in query:
        print(f"User said: {query}\n")

    except Exception as e:
        #print(e)
        var.set("Say that again please...")
        screen.update()
        print("Say that again please...")
        #speak("Say that again please...")
        return "None"
    var1.set(query)
    screen.update()
    return query.lower()
```

```
def note():
    var.set("what you want to write sir")
    screen.update()
    speak("what you want to write sir")
    write = takeCommand()
    #date =datetime.datetime.now().strftime("%H:%M:%S")
    file_name = takeCommand() + "-note.txt"

    with open(file_name, "w") as f:
        f.write(write)

    path_1 ="D:\\virtual assistant final file\\"+str(file_name)
    path_2 ="D:\\virtual assistant final file\\database\\notepad files\\"+str(file_name)

    os.rename(path_1,path_2)
    os.startfile(path_2)

    subprocess.Popen(["notepad.exe", file_name])
    var.set("notepad file saved")
    screen.update()
    speak("notepad file saved")



def date():
    now = datetime.datetime.now()
    month_name = now.month
    day_name = now.day
```

```
    month_names = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']

    ordinalnames = [ '1st', '2nd', '3rd', ' 4th', '5th', '6th', '7th', '8th', '9th', '10th', '11th', '12th', '13th', '14th', '15th', '16th', '17th', '18th', '19th', '20th', '21st', '22nd', '23rd','24rd', '25th', '26th', '27th', '28th', '29th', '30th', '31st']

    var.set("Today is "+ month_names[month_name-1] +" " + ordinalnames[day_name-1] + '.')

    screen.update()

    speak("Today is "+ month_names[month_name-1] +" " + ordinalnames[day_name-1] + '.')


#EMAIL DEF FUNCTION

def send_email(receiver, subject, message):

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

    server.ehlo()

    server.starttls()

    server.login('virtualbotmee@gmail.com','Danych@0806')

    email =EmailMessage()

    email['From'] ='virtualbotmee@gmail.com'

    email['To'] = receiver

    email['Subject'] = subject

    email.set_content(message)

    server.send_message(email)

    server.close()


def markattendance(query):

    with open('history.csv','r+') as f:

        f.readlines()

    time =datetime.datetime.now().strftime(" Hours:%H minutes:%M ")

    f = open('history.csv','a+')

    f.write(query  )
```

```
    f.write(time )
    f.write('\n')
    f.close()




def TaskExecution():
    global speak
    wishMe()


    while True:
        query = queryentry.get()
        query = takeCommand().lower()
        markattendance(query)



        if 'wikipedia' in query:
            try:
                var.set('searching wikipedia...')
                screen.update()
                speak('searching wikipedia...')
                query = query.replace("wikipedia","")
                results = wikipedia.summary(query,sentences=1)
                print(results)
                speak("According to wikipeadia ")
```

```
            var.set(results)
            screen.update()
            speak(results)

        except Exception as e:
            var.set("sorry sir, I am not able to find")
            screen.update()
            speak("sorry sir, I am not able to find")



    elif'on youtube'in query:
        song = query.replace('play','')
        var.set('playing'+ song)
        screen.update()
        speak('playing'+ song)
        kt.playonyt(song)



    elif'on google'in query:
        var.set("ok sir, searching on google")
        screen.update()
        speak("ok sir, searching on google")
        query=query.replace("mee","")
        query=query.replace("on google search","")
        kt.search(query)
        var.set("this is what i found for your search sir!!")
        screen.update()
```

```
    speak("this is what i found for your search sir!!")

elif 'website' in query:
    var.set("opening...")
    screen.update()
    speak("opening...")
    #name = takeCommand()
    query=query.replace("mee","")
    query=query.replace("website","")
    query=query.replace("open","")
    webbrowser.open( 'google.com'+ query +".com" )
    speak("here is your result")

elif'the time'in query:
    time =datetime.datetime.now().strftime("%H'Hours':%M 'minutes':%S 'seconds'")
    print(time)
    var.set(f"sir, the time is{time}")
    screen.update()
    speak(f"sir, the time is{time}")

elif'open vs code' in query:
    var.set('Opening vs code')
    screen.update()
    speak('Opening vs code')
    codePath = "D:\\Microsoft VS Code\\Code.exe"
    os.startfile(codePath)

elif 'map' in query:
```

```
            query = query.replace('search',"")
            query = query.replace('on map',"")
            var.set("okay sir searching for" + query)
            screen.update()
            speak("okay sir searching for" + query)
            speak('')
            webbrowser.open("https://www.google.com/maps/place/" + query)


    # CLOSING FUNCTIONS
        elif'close vs code' in query:
            var.set('closing vs code')
            screen.update()
            speak('closing vs code')
            os.system("TASKKILL /F /im Code.exe")

        elif'close youtube' in query:
            var.set("closing youtube sir!!")
            screen.update()
            speak("closing youtube sir!!")
            os.system("TASKKILL /F /im chrome.exe")

        elif'close google' in query:
            var.set("closing google sir!!")
            screen.update()
            speak("closing google sir!!")
            os.system("TASKKILL /F /im chrome.exe")
```

```
        elif'close wikipedia' in query:
            var.set("closing wikipedia sir!!")
            screen.update()
            speak("closing wikipedia sir!!")
            os.system("TASKKILL /F /im chrome.exe")

    # EMAIL FUNCTION
        elif 'send email' in query:
            try:
                var.set("To whom you want to send email")
                screen.update()
                speak("To whom you want to send email")
                name=takeCommand()
                receiver = mailDict[name]
                var.set("what is the subject of the mail")
                screen.update()
                speak("what is the subject of the mail")
                subject = takeCommand()
                var.set("what is the message or text of the mail")
                screen.update()
                speak("what is the message or text of the mail")
                message = takeCommand()
                send_email(receiver, subject, message)
                var.set("Email has been sent!")
                screen.update()
                speak("Email has been sent!")
            except Exception as e:
                print(e)
```

```
            var.set("sorry sir, I am not able to send this email")
            screen.update()
            speak("sorry sir, I am not able to send this email")

# JOKE FUNCTION
        elif 'joke' in query:
            var.set(pyjokes.get_joke())
            screen.update()
            speak(pyjokes.get_joke())

# ASSISTANT CLOSE FUNCTION
        elif 'goodbye me' in query:
            var.set('ok sir!! goodbye')
            screen.update()
            speak('ok sir!! goodbye')
            sys.exit()

        elif 'take a break' in query:
            var.set("ok sir, you can call me any time")
            screen.update()
            speak("ok sir, you can call me any time")
            break
# SYSYTEM RESTART FUNCTION
        elif 'restart pc' in query:
            var.set("Do you want to restart")
            screen.update()
            speak("Do you want to restart")
            query= takeCommand()
```

```
            if 'yes' in query:
                var.set("system is restarting")
                screen.update()
                speak("system is restarting")
                os.system('shutdown /r /t 0')
            else:
                var.set("sorry sir,I am not able to hear any response about restar pc ")
                screen.update()
                speak("sorry sir,I am not able to hear any response about restar pc ")
    # BASIC FUNCTIONS
        elif "hello" in query:
            var.set("Hello sir, i am mee your assistant   how may i help you")
            screen.update()
            speak("Hello sir, i am mee your assistant   how may i help you")

        elif 'how are you' in query:
            var.set("i am fine sir")
            screen.update()
            speak("i am fine sir")
            var.set("what about you")
            screen.update()
            speak("what about you")

            about = takeCommand()
            if "fine" in about:
                var.set("it is great sir i am glad to hear that")
                screen.update()
                speak("it is great sir i am glad to hear that")
```

```
    elif "good" in about:
        var.set("it is great sir i wish you have a great day ahead")
        screen.update()
        speak("it is great sir i wish you have a great day ahead")
    elif "not well" in about:
        var.set("what happend sir is that everything is fine")
        screen.update()
        speak("what happend sir is that everything is fine")
        query=takeCommand()
        if "lost" in query:
            var.set("sorry for your loss sir!!")
            screen.update()
            speak("sorry for your loss sir!!")
        else:
            var.set("ok sir,  dont worry, everything will be fine, i am with you ")
            screen.update()
            speak("ok sir,  dont worry")
            speak("everything will be fine")
            speak("i am with you")

elif "hey are you there" in query:
    var.set("yes sir, i am always here for you")
    screen.update()
    speak("yes sir, i am always here for you")
    var.set("what can i do for you sir!!")
    screen.update()
    speak("what can i do for you sir!!")
```

```
# SCREENSHOT FUNCTION
        elif 'take a screenshot' in query:
            var.set('ok sir what should be the name of file')
            screen.update()
            speak('ok sir what should be the name of file')
            name1= takeCommand()
            name= name1 + '.png'
            path = 'D:\\virtualbotss\\ss' + name
            ss= pyautogui.screenshot()
            ss.save(path)
            var.set('screenshot has been save')
            screen.update()
            speak ('screenshot has been save')

# clear history
        elif 'clear search history' in query:
            f = open('history.csv','w')
            f.truncate()
            f.close()
            var.set('search history is cleared sir')
            screen.update()
            speak('search history is cleared sir')

# show history
        elif 'show search history' in query:
            var.set('showing history sir')
            screen.update()
            speak('showing history sir')
```

```
            f = open('history.csv','r')
            text = f.read ()
            var.set(text)
            screen.update()
            print(text)
            f.close()


# global dictionary
        elif 'meaning'in query:
            query = query.replace('what is','')
            query = query.replace('the meaning','')
            query = query.replace('of','')
            result= PyDictionary.meaning(query)
            var.set(f"the meaning for{query} is {result}")
            screen.update()
            speak(f"the meaning for{query} is {result}")

        elif 'synonym' in query:
            query = query.replace('what is','')
            query = query.replace('the synonym','')
            query = query.replace('of','')
            result= PyDictionary.synonym(query)
            var.set(f"the synonym for{query} is {result}")
            screen.update()
            speak(f"the synonym for{query} is {result}")

        elif "date" in query:
```

```
            date()



#notes
        elif "take notes" in query:
            note()

def main():
    while True:
        query = queryentry.get()
        markattendance(query)
        if "exit" in query:
            break
        if "wikipedia"in query:
            var.set("ok sir searching ")
            screen.update()
            speak("ok sir searching ")
            query= query.replace("wikipedia","")
            result= wikipedia.summary(query,sentences=2)
            var.set(result)
            screen.update()
            speak(result)

        elif"google"in query:
            var.set("ok sir searching ")
            screen.update()
            speak("ok sir searching ")
            query= query.replace("google","")
```

```
        result = kt.search(query)
        var.set(result)
        screen.update()
        speak(result)

    elif "date" in query:
        date()

    elif 'meaning'in query:
        query = query.replace('what is','')
        query = query.replace('the meaning','')
        query = query.replace('of','')
        result= PyDictionary.meaning(query)
        var.set(f"the meaning for{query} is {result}")
        screen.update()
        speak(f"the meaning for{query} is {result}")

    elif 'synonym' in query:
        query = query.replace('what is','')
        query = query.replace('the synonym','')
        query = query.replace('of','')
        result= PyDictionary.synonym(query)
        var.set(f"the synonym for{query} is {result}")
        screen.update()
        speak(f"the synonym for{query} is {result}")


    elif'on youtube'in query:
```

```
            song = query.replace('play','')
            var.set('playing'+ song)
            screen.update()
            speak('playing'+ song)
            kt.playonyt(song)

        elif "Hello" in query:
            var.set("Hello sir")
            speak("Hello sir")



def here():
    speak("type search is activated")



#>>>>>>>>>>>>>>>>>>>>>>>>>>>SETTING
FUCTION<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

def change_name():

  name_info = name.get()

  file=open("Assistant_name", "w")

  file.write(name_info)

  file.close()
```

```
  settings_screen.destroy()

  screen.destroy()


def change_name_window():

    global settings_screen
    global name


    settings_screen = Toplevel(screen)
    settings_screen.title("Settings")
    settings_screen.geometry("400x500")
    settings_screen.iconbitmap('20.ico')


    name = StringVar()

    current_label = Label(settings_screen, text = "  Current name: "+ name_assistant)
    current_label.grid()

    enter_label = Label(settings_screen, text = "  Please enter your Virtual Assistant's name
below")
    enter_label.grid(row=1,column=0)


    Name_label = Label(settings_screen, text = "  Name")
```

```
    Name_label.grid(row=2,column=0)


    name_entry = Entry(settings_screen, textvariable = name)
    name_entry.grid()



    change_name_button = Button(settings_screen, text = "Ok", width = 10, height = 1,
command = change_name)
    change_name_button.grid()




#>>>>>>>>>>>>>>>>>>>>>>>>>>>INFORMATION ABOUT THE
APPLICATION<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
def info():


  info_screen = Toplevel(screen)
  info_screen.title("Info")
  info_screen.geometry("800x900")
  info_screen.iconbitmap('20.ico')


  creator_label = Label(info_screen,text = "Created by Bhavik Sutar")
  creator_label.grid()


  A_label = Label(info_screen, text= " User can  easy to do voice search an also able to search by
type by activating type search  ")
  A_label.grid()


  B_label = Label(info_screen, text = "The features of the application: ")
```

```
  B_label.grid()


  C_label = Label (info_screen, text="(1): To search, open or perform web task user must want to mention thise words in the query \n 1. For Wikipedida 'on wikipedia' \n 2. For Google 'on google' \n 3.For Time 'the time'    \n 4. For searching for location on map 'map' \n 5. For Date 'date'      " )

  C_label.grid()


  D_label = Label (info_screen, text="(2): To ask basic questions user need to mention thise words in the query \n 1. For Hello 'hello' \n 2. Foe How are you 'how are you' \n 3. To check assistant is active or not 'hey are you there' \n 4. To meaning 'meaning' \n 5. To synonym of any word 'synonym' \n 6. For Jokes 'joke'" )

  D_label.grid()


  E_label = Label(info_screen, text="(3): To perform system task user need to mention thise words in query \n 1. For Screen shot 'take screenshot \n 2.To Make Notes 'note' \n 3. To Restar PC 'restart pc' \n 4. To Send Mail 'send email' \n 5. To Open VS Code 'open Vs code'")

  E_label.grid()


  F_label = Label(info_screen, text="(4): To close the web applications user need to call the 'close' and Application name" )

  F_label.grid()



keyboard.add_hotkey("F4", TaskExecution)


def wikipedia_screen(text):



  wikipedia_screen = Toplevel(screen)

  wikipedia_screen.title(text)

  wikipedia_screen.iconbitmap('20.ico')
```

```
  message = Message(wikipedia_screen, text= text)
  message.grid()



screen.title(name_assistant)


screen.geometry("1200x1500")
screen.maxsize(1400,1600)
screen.iconbitmap('20.ico')
screen.configure(bg= 'sky blue' )



def stop():
    global run
    run= False


#name_label = Label(text = name_assistant,width = 300, bg = "black", fg="white", font =
("Calibri", 13))
#name_label.grid()


label1 = Label(screen, textvariable = var, bg = '#ADD8E6')
label1.config(font=("Courier", 10))
var.set('Welcome')
label1.place(x=0,y=50)


label2 = Label(screen, textvariable = var1, bg = '#FAB60C')
label2.config(font=("Courier", 10))
var1.set('User Said:')
```

```
label2.place(x=0,y=100)

background_image =PhotoImage(file="last.png")
background_label = Label(screen, image=background_image, )
background_label.place(x=0,y=200)

microphone_photo = PhotoImage(file = "assistant_logo.png")
microphone_button = Button(image=microphone_photo, command =
threading.Thread(target=TaskExecution).start)
microphone_button.place(x=450,y=300)

settings_photo = PhotoImage(file = "settings.png")
settings_button = Button(image=settings_photo, command = change_name_window)
settings_button.place(x=550,y=700)

stop =  Button(screen, text="STOP",command=screen.destroy)
stop.place(x=435,y=700)

info_button = Button(text ="INFO", command =info)
info_button.place(x=700,y=700)

#Button(screen,text=" TYPE ",command=
threading.Thread(target=here).run).place(x=560,y=600)



queryentry = Entry(screen, width=40)
queryentry.place(x=400,y=500)
queryentry.delete(0,10)
```

```
Button(screen,text="SEARCH", command= main).place(x=560,y=550)


screen.mainloop()
#main_screen()
```

➢ BIBLIOGRAPHY

1. Kahn, D., The Codebreakers, Macmillan Publishing Company, New York, 1967.

2. Diffie, W., and Hellman, M.E., “Privacy and Authentication: An Introduction to Cryptography,” Proc. IEEE, vol. 67, no. 3, Mar. 1979, pp. 397–427.

3. Beker ,H., and Piper, F., Cipher Systems, John Wiley & Sons, Inc., New York, 1982.

4. Denning, D.E.R., Cryptography and Data Security, Addison-Wesley Publishing Company, Reading, Mass., 1982.

5. Shannon, C.E., “Communication Theory of Secrecy Systems,” Bell Syst. Tech. J., vol. 28, Oct. 1949, pp. 656–715.

6. Hellman, M. E., “An Extension of the Shannon Theory Approach to Cryptography,” IEEE Trans. Inf. Theory, vol. IT23, May 1978, pp. 289–294.

7. Smith, J. L., “The Design of Lucifer, a Cryptographic Device for Data Communications,” IBM Research Rep. RC-3326, 1971.

8. Feistel, H. “Cryptography and Computer Privacy,” Sci. Am., vol. 228, no. 5, May 1973, pp. 15–23.

9. National Bureau of Standards, “Data Encryption Standard,” Federal Information Processing Standard (FIPS), Publication no. 46, Jan. 1977.

10. United States Senate Select Committee on Intelligence, “Unclassified Summary: Involvement of NSA in the Development of the Data Encryption Standard,” IEEE Common. Soc. Mag., vol. 16, no. 6, Nov. 1978, pp. 53–55.

11. Stallings, W., Cryptography and Network Security, Second Edition, Prentice Hall, Upper Saddle River, NJ. 1998.

12. Diffie, W., and Hellman, M. E., “New Directions in Cryptography,” IEEE Trans. Inf. Theory, vol. IT22, Nov. 1976, pp. 644–654.

13. Rivest, R.L., Shamir, A., and Adelman, L., “On Digital Signatures and Public Key Cryptosystems,” Common. ACM, vol. 21, Feb. 1978, pp. 120–126.