

KENDRIYA VIDYALAYA MALLESWARAM



PROJECT TITLE

JARVIS (AN AI ASSISTANT)

(COMPUTER SCIENCE PROJECT)

CLASS XII B - GROUP : 17

NAME	ROLL NO.	CBSE ROLL NO.
VENKAT NARAYAN TL	12251	-

CERTIFICATE

This is to certify that **VENKAT NARAYAN TL** of class **XII B** have successfully completed the Computer Science project.

The project titled “**JARVIS (AN AI ASSISTANT)**” is the bonafide work of the above students as per the CBSE curriculum 2022-23.

INTERNAL EXAMINER

PRINCIPAL

EXTERNAL EXAMINER

[BHAVANA]

PGT — COMPUTER SCIENCE

ACKNOWLEDGEMENT

We would like to take this opportunity to express our sincere gratitude to our computer science teacher Ms. BHAVANA for all her support and encouragement, crucial to our completion of this project.

We would also like to thank our parents and friends who helped us a lot in coming up with the idea, making it into a viable project and giving their inputs all the way to help us bring this to fruition and test it. Finally, we would like to thank our principal Mrs. H.D. BHANUMATHY for her constant support.

TABLE OF CONTENTS

S. NO.	CONTENTS
1	INTRODUCTION
2	PROJECT SETUP
3	SOURCE CODE
4	OUTPUT
5	FUTURE SCOPE
6	BIBLIOGRAPHY
7	NOTES

1. INTRODUCTION

In this modern era, day to day life became smarter and interlinked with technology. We already know some voice assistants like Google, Siri.etc. Now in our voice assistants system, it can write notes in notepad, calculate and remember what we said by remember command.

This project works through voice input and gives output through voice and displays the text on the screen. The main agenda of our voice assistants makes people smart and give instant and computed results. The voice assistants takes the voice input through our microphone (Bluetooth and wired microphone) and it converts our voice into computer understandable language gives the required solutions and answers which are asked by the user. This assistant connects with the World Wide Web to provide results that the user has questioned. Natural Language processing algorithm helps computer machines to engage in communication using natural human language in many forms.

Today the developments of artificial intelligence (AI) systems that can organize a natural human-machine interaction (through voice, communication, gestures, facial expressions, etc.) are gaining in popularity. One of the most studied and popular was the direction of interaction, based on the understanding of the machine by the machine of the natural human language. It is no longer a human who learns to communicate with a machine, but a machine learns to communicate with a human, exploring his actions, habits, behaviour and trying to become his personalized assistant.

This system is designed to be used efficiently on desktops. Personal assistants software improves user productivity by managing routine tasks of the user and by providing information from an online source to the user.

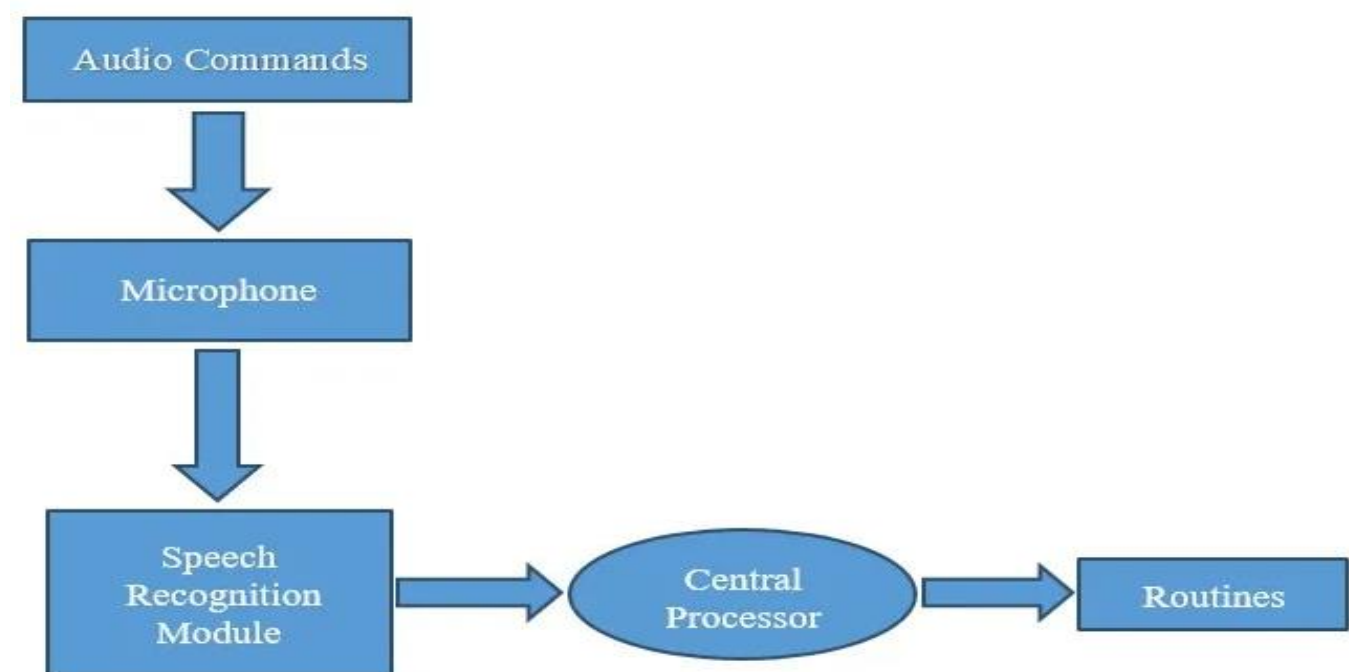
This project was started on the premise that there is a sufficient amount of openly available data and information on the web that can be utilized to build a virtual assistant that has access to making intelligent decisions for routine user activities.

Each company developer of the intelligent assistant applies his own specific methods and approaches for development, which in turn affects the final product. One assistant can synthesize speech more qualitatively, another can more accurately and without additional explanations and corrections perform tasks, others can perform a narrower range of tasks, but most accurately and as the user wants. Obviously, there is no universal assistant who would perform all tasks equally well. The set of characteristics that an assistant has depends entirely on which area the developer has paid more attention to. Since all systems are based on machine learning methods and use for their creation huge amounts of data collected from various sources and then trained on them, an important role is played by the source of this data, be it search systems, various information sources or social networks. The amount of information from different sources determines the nature of the assistant, which can result as a result. Despite the different approaches to learning, different algorithms and techniques, the principle of building such systems remain approximately the same.

Figure shows the technologies that are used to create intelligent systems of interaction with a human by his natural language. The main technologies are voice activation, automatic speech recognition, Text-To-Speech, voice biometrics, dialogue manager, natural language understanding and named entity recognition.

Voice Technology	Brain Technology
Voice Activation	Voice Bio-metrics
Automatic Speech Recognition (ASR)	Dialog Management
(Teach-To-Speech (TTS)	Natural Language Understanding (NLU)
	Named Entity Recognition NER)

BASIC WORKFLOW



2. PROJECT SETUP

For coding this project we need to install the following modules and external libraries:

pyttsx3

pyttsx3 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. To install this module, type the below command in the terminal or command prompt:

```
C:\Users\Venkat>pip install pyttsx3
```

SpeechRecognition

This allows us to convert audio into text for further processing.

To install this module, type the below command in the terminal:

```
C:\Users\Venkat>pip install SpeechRecognition
```


pywhatkit

This is an easy-to-use library that will help us interact with the browser very easily. To install the module, run the following command in the terminal or command prompt:

```
C:\Users\Venkat>pip install pywhatkit
```

webbrowser

This provides a high level interface to allow displaying web based documents to users. To install this module run the following command in command prompt or terminal:

```
C:\Users\Venkat>pip install webbrowser
```

wikipedia

We'll use this to fetch a variety of information from the Wikipedia website. To install this module, type the below command in the terminal or command prompt:

```
C:\Users\Venkat>pip install wikipedia
```

os

It is an in-built module that provides a way of using operating system-dependent functionality, enabling us to form a link with the operating system.

Datetime

The datetime module provides us with classes for manipulating dates and times. For example, we can get the current timestamp using this module.

```
C:\Users\Venkat>pip install datetime
```

Requests

The requests module has several built-in methods to make HTTP requests to a specified URL. We can use the HTTP methods GET, POST, PUT, PATCH, and HEAD with the requests. An HTTP request enables us to retrieve data from a specified URL as well as push data to a server.

```
C:\Users\Venkat>pip install requests
```

Pyjokes

The pyjokes library lets us generate one-liner programming jokes. A new joke is generated every time. As we want JARVIS to generate programming jokes for us, this library will come in handy.

```
C:\Users\Venkat>pip install pyjokes
```

Tkinter

Tkinter is a GUI library in Python that is a fast and easy way to create GUI applications.

```
C:\Users\Venkat>pip install tkinter
```

bs4

Beautiful Soup is a Python library for pulling data out of HTML and XML files.

```
C:\Users\Venkat>pip install bs4
```

pywikihow

An unofficial WikiWow python API. Uses BeautifulSoup to scrape WikiHow and return the data you want.

```
C:\Users\Venkat>pip install pywikihow
```

pyautogui

Python package that works across Windows, MacOS X and Linux which provides the ability to simulate mouse cursor moves and clicks as well as keyboard button presses.

```
C:\Users\Venkat>pip install pyautogui
```

Operator

The Python operator module is one of the inbuilt modules in Python, and it provides us with a lot of functions such as `add(x, y)`, `floordiv(x, y)` etc., which we can use to perform various mathematical, relational, logical and bitwise operations on two input numbers.

3. SOURCE CODE

IMPORTING THE MODULES

```
import os
from tkinter import *
import pyttsx3
import speech_recognition as sr
import webbrowser
import pywhatkit
import wikipedia
import pyautogui
import pyjokes
from datetime import datetime, date
import requests
from bs4 import BeautifulSoup
from pywikihow import search_wikihow
import operator
```

CREATING A WINDOW FOR GUI

```
window = Tk()

global var
global var1
global var2
global var3

var = StringVar()
var1 = StringVar()
var2 = StringVar()
var3 = StringVar()
```

SPEAK FUNCTION

```
def Speak(audio):
    engine = pyttsx3.init('sapi5')
    voices = engine.getProperty('voices')
```

```
engine.setProperty('voice', voices[0].id)
engine.setProperty('rate', 170)
```

```
print(" ")
print(f"JARVIS SAID : {audio}")
engine.say(audio)
engine.runAndWait()
#print(f" : {audio}")
print(" ")
```

Speak("Initializing Jarvis")

TAKING INPUT FROM THE USER

```
def takecommand():
    command = sr.Recognizer()
    with sr.Microphone() as source:
        var1.set("LISTENING....")
        window.update()
        print("LISTENING....")
        command.pause_threshold = 1
        audio = command.listen(source, 0, 2)

    try:
        var1.set("RECOGNIZING.....")
        window.update()
        print("RECOGNIZING.....")
        query = command.recognize_google(audio, language='en-in')
        print(f"YOU SAID : {query}")

    except:
        return "none"

    var2.set(query)
    window.update()

    return query.lower()
```

TASK FUNCTION FOR JARVIS

```
def TaskExe():
    #Speak("Hello sir")
```

```

def wishme():
    hour = int(datetime.now().hour)
    tt = datetime.now().strftime("%I:%M %p")

    if hour>=0 and hour<12:
        var1.set(f"Good morning Sir, You called me at {tt}")
        window.update()
        Speak(f"Good morning Sir, You called me at {tt}")

    elif hour>12 and hour<17:
        var1.set(f"Good afternoon sir, You called me at {tt}")
        window.update()
        Speak(f"Good afternoon sir, You called me at {tt}")

    else:
        var1.set(f"Good evening sir, You called me at {tt}")
        window.update()
        Speak(f"Good evening sir, You called me at {tt}")

wishme()
var1.set("I'm Jarvis here , How may I help you sir?")
window.update()
Speak("I'm Jarvis here , How may I help you sir?")

def OpenApps():
    var1.set("Ok sir, wait a second!")
    window.update()
    Speak("Ok sir, wait a second!")

    if 'facebook' in query:
        webbrowser.open("https://www.facebook.com/")

    elif 'chrome' in query:
        os.startfile("C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe")

    elif 'instagram' in query:
        webbrowser.open("https://www.instagram.com/")

    elif 'maps' in query:
        webbrowser.open("https://www.google.co.in/maps/@12.9728512,77.6077312,12z")

    elif 'youtube' in query:
        webbrowser.open("https://www.youtube.com/")

    Speak("Your app has been opened sir!")

def CloseApp():
    var1.set("Sure sir!")
    window.update()
    Speak("Sure sir!")

```



```
if 'youtube' in query:  
    os.system("TASKKILL /F /im chrome.exe")
```

```
elif 'chrome' in query:  
    os.system("TASKKILL /f /im chrome.exe")
```

```
elif 'facebook' in query:  
    os.system("TASKKILL /F /im chrome.exe")
```

```
elif 'instagram' in query:  
    os.system("TASKKILL /F /im chrome.exe")
```

```
elif 'maps' in query:  
    os.system("TASKKILL /F /im chrome.exe")
```

```
Speak("Your app has been closed sir!")
```

```
def screenshot():  
    var1.set("Ok sir , what should I name that file")  
    window.update()  
    Speak("Ok sir , what should I name that file")  
    path = takecommand()  
    path1name = path + ".png"  
    path1 = "C:\\Users\\Venkat\\OneDrive\\Pictures\\Screenshots\\" + path1name  
    kk = pyautogui.screenshot()  
    kk.save(path1)  
    os.startfile("C:\\Users\\Venkat\\OneDrive\\Pictures\\Screenshots\\")  
    var1.set("here is your screenshot sir!")  
    window.update()  
    Speak("here is your screenshot sir!")
```

```
def Temp():  
    search = "temperature in bangalore"  
    url = f"https://www.google.com/search?q={search}"  
    r = requests.get(url)  
    data = BeautifulSoup(r.text, "html.parser")  
    temperature = data.find("div", class_="BNeawe").text  
    var1.set(f"The Temperature Outside Is {temperature}")  
    window.update()  
    Speak(f"The Temperature Outside Is {temperature}")
```

```
var1.set("Do I Have To Tell You Another Place Temperature ?")  
window.update()  
Speak("Do I Have To Tell You Another Place Temperature ?")  
next = takecommand()
```

```
if 'yes' in next:
```

```
    var1.set("can you please name the place sir ")
```

```
window.update()
Speak("can you please name the place sir ")
name = takecommand()
search = f"temperature in {name}"
url = f"https://www.google.com/search?q={search}"
r = requests.get(url)
data = BeautifulSoup(r.text, "html.parser")
temperature = data.find("div", class_="BNeawe").text
var1.set(f"The Temperature in {name} is {temperature}")
window.update()
Speak(f"The Temperature in {name} is {temperature}")
```

else:

```
var1.set("Thank You sir")
window.update()
Speak("Thank You sir")
```

def Notepad():

```
var1.set("Tell me query sir!")
window.update()
Speak("Tell me query sir!")
var1.set("I'm ready to write.")
window.update()
Speak("I'm ready to write.")
```

```
writes = takecommand()
```

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

```
filename = str(time).replace(":", "-") + "-note.txt"
```

```
with open(filename, "w") as file:
```

```
    file.write(writes)
```

```
path_1 = "C:\\Users\\Venkat\\PycharmProjects\\Jarvis\\" + str(filename)
```

```
path_2 = "C:\\Users\\Venkat\\PycharmProjects\\Jarvis\\Notepad saving from python  
automation\\" + str(filename)
```

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

```
os.startfile(path_2)
```

def Daydateandtime():

```
day = datetime.today().weekday() + 1
```

```
day_dict = {1: 'Monday', 2: 'Tuesday', 3: 'Wednesday', 4: 'Thursday', 5: 'Friday', 6: 'Saturday', 7:  
'Sunday'}
```

```
today = date.today()
```

```
datec = today.strftime("%B %d, %Y")
```

```
ctime = datetime.now().strftime("%I:%M %p")
```

```
if day in day_dict.keys():
```

```
    d = day_dict[day]
```

```
print(d)
print(datec)
var1.set("Time is " + ctime)
window.update()
Speak("Time is " + ctime)
var1.set("The day is " + d + " and the date is " + datec)
window.update()
Speak("The day is " + d + " and the date is " + datec)
```

TAKING COMMAND FROM THE USER AND ANSWERING TO THE QUESTION THE USER ASKS

while True:

```
    query = takecommand()
```

if "hello" in query or 'hey' in query:

```
    var1.set("Hello sir, I am Jarvis")
    window.update()
    Speak("Hello sir, I am Jarvis")
    var1.set("Your personal AI assistant!")
    window.update()
    Speak("Your personal AI assistant!")
    var1.set("How may I help you?")
    window.update()
    Speak("How may I help you?")
```

elif "how are you" in query:

```
    var1.set("I am good Sir! What about You?")
    window.update()
    Speak("I am good Sir! What about You?")
```

elif "i am good" in query or 'i am fine' in query:

```
    var1.set("That's nice sir")
    window.update()
    Speak("That's nice sir")
    var1.set("How can i help you sir?")
    window.update()
    Speak("How can i help you sir?")
```

elif "tell me about yourself" in query or 'about you' in query:

```
    var1.set("I am Jarvis an AI assistant")
    window.update()
    Speak("I am Jarvis an AI assistant")
    var1.set("created by Venkat and Prathuysh ")
    window.update()
    Speak("created by Venkat and Prathuysh ")
```

elif "you need a break" in query:

```
var1.set("Thankyou Sir, I'm leaving now, you can call me anytime! ")
window.update()
Speak("Thankyou Sir, I'm leaving now, you can call me anytime! ")
exit()
```

elif 'good bye' in query or 'goodbye' in query or 'bye' in query:

```
var1.set(" Thank you sir, I'm leaving now, bye!")
window.update()
Speak(" Thank you sir, I'm leaving now, bye!")
exit()
```

elif 'youtube search' in query:

```
var1.set("Ok sir, This is what i found for your search")
window.update()
Speak("Ok sir, This is what i found for your search")
query = query.replace("jarvis", "")
query = query.replace("youtube search", "")
web = 'https://www.youtube.com/results?search_query=' + query
webbrowser.open(web)
var1.set("Done sir!")
window.update()
Speak("Done sir!")
```

elif 'website' in query:

```
var1.set("Ok sir, Launching.....")
window.update()
Speak("Ok sir, Launching.....")
query = query.replace("jarvis", "")
query = query.replace("website", "")
query = query.replace(" ", " ")
web1 = query.replace("open", "")
web2 = "https://www." + web1 + ".com"
webbrowser.open(web2)
var1.set("Launched sir!")
window.update()
Speak("Launched sir!")
```

elif 'launch' in query:

```
var1.set("can you please tell me the name of the website sir!")
window.update()
Speak("can you please tell me the name of the website sir!")
name = takecommand()
web = "https://www." + name + ".com"
webbrowser.open(web)
var1.set("Done sir!")
window.update()
Speak("Done sir!")
```

elif 'wikipedia' in query:

```
var1.set("Searching wikipedia....")
window.update()
```

```
Speak("Searching wikipedia....")
query = query.replace("jarvis", "")
query = query.replace("wikipedia", "")
wiki = wikipedia.summary(query, 2)
var1.set(f"According to wikipedia : {wiki}")
window.update()
Speak(f"According to wikipedia : {wiki}")
```

```
elif 'screenshot' in query:
    screenshot()
```

```
elif 'open facebook' in query:
    OpenApps()
```

```
elif 'open chrome' in query:
    OpenApps()
```

```
elif 'open instagram' in query:
    OpenApps()
```

```
elif 'open maps' in query:
    OpenApps()
```

```
elif 'open youtube' in query:
    OpenApps()
```

```
elif 'close youtube' in query:
    CloseApp()
```

```
elif 'close chrome' in query:
    CloseApp()
```

```
elif 'close maps' in query:
    CloseApp()
```

```
elif 'close facebook' in query:
    CloseApp()
```

```
elif 'close instagram' in query:
    CloseApp()
```

```
elif 'joke' in query:
    get = pyjokes.get_joke()
    var1.set(get)
    window.update()
    Speak(get)
```

```
elif 'nice' in query:
    var1.set('Thank you sir')
    window.update()
    Speak('Thank you sir')
```

elif 'repeat my words' in query:

```
var1.set("Sure sir!")
window.update()
Speak("Sure sir!")
jj = takecommand()
var1.set(f"You said : {jj}")
window.update()
Speak(f"You said : {jj}")
```

elif 'my location' in query:

```
var1.set("Ok sir, wait a second")
window.update()
Speak("Ok sir, wait a second")
webbrowser.open('https://www.google.co.in/maps/@12.9728512,77.6077312,12z')
```

elif 'alarm' in query:

```
var1.set("Enter the time in command prompt!")
window.update()
Speak("Enter the time !")
time = input(": Enter the time :")
```

while True:

```
Time_Ac = datetime.now()
now = Time_Ac.strftime("%H:%M:%S")
```

if now == time:

```
var1.set("Time Up sir!")
window.update()
Speak("Time Up sir!")
```

elif now > time:

```
break
```

elif 'remember that' in query:

```
remembermsg = query.replace("remember that", "")
remembermsg = remembermsg.replace("jarvis", "")
var1.set(f"Sir please tell me to remind you that : " + remembermsg)
window.update()
Speak(f"Sir please tell me to remind you that : " + remembermsg)
remember = open('data.txt', 'w')
remember.write(remembermsg)
remember.close()
```

elif 'what do you remember' in query:

```
remember = open('data.txt', 'r')
Speak("You told me that" + remember.read())
```

elif 'google search' in query:

```
import wikipedia as googleScrap
query = query.replace("jarvis", "")
```

```
query = query.replace("google search", "")
query = query.replace("google", "")
var1.set("This is what I found for your search")
window.update()
Speak("This is what I found for your search")
pywhatkit.search(query)
```

try:

```
result = googleScrap.summary(query, 3)
var1.set(result)
window.update()
Speak(result)
```

except:

```
var1.set("No data found!")
window.update()
Speak("No data found!")
```

elif 'how to' in query:

```
var1.set("Getting data from internet!")
window.update()
Speak("Getting data from internet!")
op = query.replace("jarvis", "")
max_result = 1
how_to_fun = search_wikihow(op, max_result)
assert len(how_to_fun) == 1
how_to_fun[0].print()
var1.set(how_to_fun[0].summary)
window.update()
Speak(how_to_fun[0].summary)
```

elif 'write a note' in query:

```
Notepad()
```

elif 'temperature' in query:

```
Temp()
```

elif 'day' in query:

```
Daydateandtime()
```

elif 'date' in query:

```
Daydateandtime()
```

elif 'time' in query:

```
Daydateandtime()
```

elif 'pause' in query:

```
pyautogui.press("k")
var1.set("video paused")
window.update()
Speak("video paused")
```

elif 'play' in query:

```
pyautogui.press("k")
var1.set("video played")
window.update()
Speak("video played")
```

elif 'mute' in query:

```
pyautogui.press("m")
var1.set("video muted")
window.update()
Speak("video muted")
```

elif 'unmute' in query:

```
pyautogui.press("m")
var1.set("video unmuted")
window.update()
Speak("video unmuted")
```

elif 'volume up' in query or 'increase' in query:

```
from keyboard import volumeup
var1.set("Turning volume up sir")
window.update()
Speak("Turning volume up sir")
volumeup()
```

elif 'volume down' in query or 'decrease' in query:

```
from keyboard import volumedown
var1.set("Turning volume down sir")
window.update()
Speak("Turning volume down sir")
volumedown()
```

elif 'sleep' in query:

```
var1.set("Going to sleep sir")
window.update()
Speak("Going to sleep sir")
exit()
```

elif 'calculate' in query:

```
command = sr.Recognizer()
with sr.Microphone() as source:
    var1.set("What do you want to calculate sir")
    window.update()
    Speak("What do you want to calculate sir")
    var1.set("LISTENING....")
    window.update()
    print("LISTENING....")
    command.pause_threshold = 1
    audio = command.listen(source, 0, 2)
    my_str = command.recognize_google(audio)
```



```

def getop(op):
    return {
        '+': operator.add,
        '-': operator.sub,
        'x': operator.mul,
        'divided': operator.__truediv__
    }[op]
def eval(op1,oper,op2):
    op1,op2 = int(op1), int(op2)
    return getop(oper)(op1, op2)
var1.set("Your result is....")
window.update()
Speak("Your result is....")
Speak(eval(*(my_str.split()))))

```

GUI INTERFACE FOR JARVIS

```

def update(ind):

    window.after(100,update,ind)

label2 = Label(window,textvariable = var,bg = 'green',foreground = 'white',padx=1120,pady=5)
label2.config(font=('ROCKWELL bold',25))
var.set("WELCOME")
label2.pack()

label3 = Label(window,textvariable = var1,bg='white',foreground = 'black',padx=700,pady=150)
label3.config(font=('ROCKWELL bold',15))
var1.set("JARVIS SAID : ")
label3.pack()

label = Label(window,textvariable=var2, bg='black',foreground = 'white',padx=1120,pady=75)
label.config(font=('ROCKWELL bold',20))
var2.set("USER SAID : ")
label.pack()

label4 = Label(window,textvariable = var3,padx=120)
label4.config(font=('ROCKWELL bold',1))
var3.set(" ")
label4.pack()

window.title('JARVIS')
window.geometry("1700x1000")

btn1 = Button(text = "CALL JARVIS",width = 20, command = TaskExe, bg = 'yellow',padx = 112,pady =

```

```
12,justify = 'left')
btn1.config(font = ('Rockwell extra bold',12))
btn1.pack()

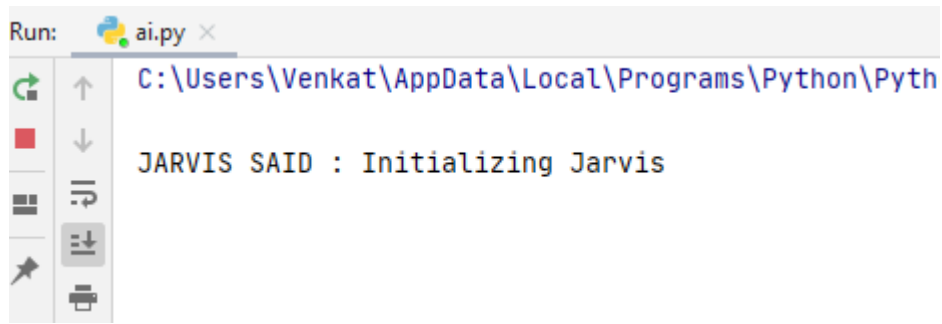
label3 = Label(window,textvariable = var3,padx=1120,pady=0)
label3.config(font=('TIMES NEW ROMAN',20))
var3.set(" ")
label3.pack()

btn2 = Button(text = "CLOSE",width = 20, command = window.destroy, bg = 'red',foreground =
'white',padx = 112,pady=12,justify='right')
btn2.config(font = ('Rockwell extra bold',12))
btn2.pack()

window.mainloop()
exit(0)
```

4. OUTPUT

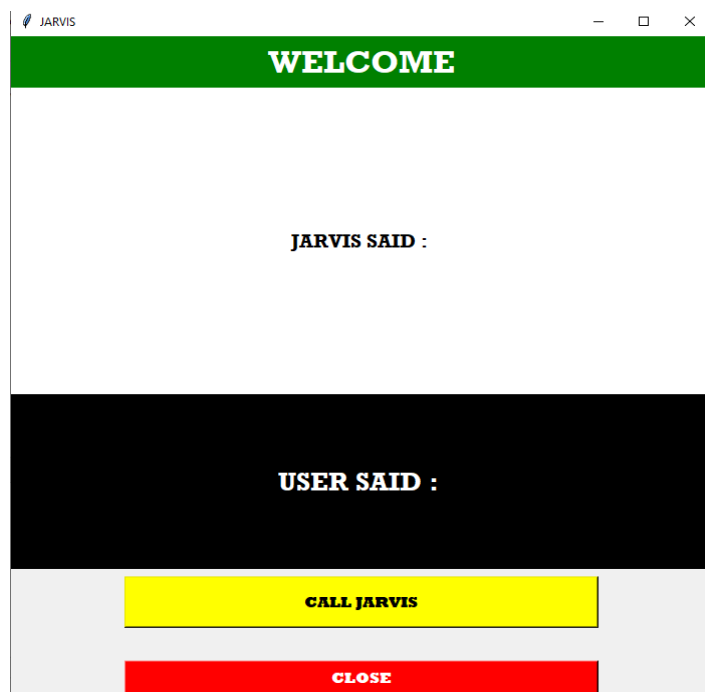
JARVIS INITIALIZATION



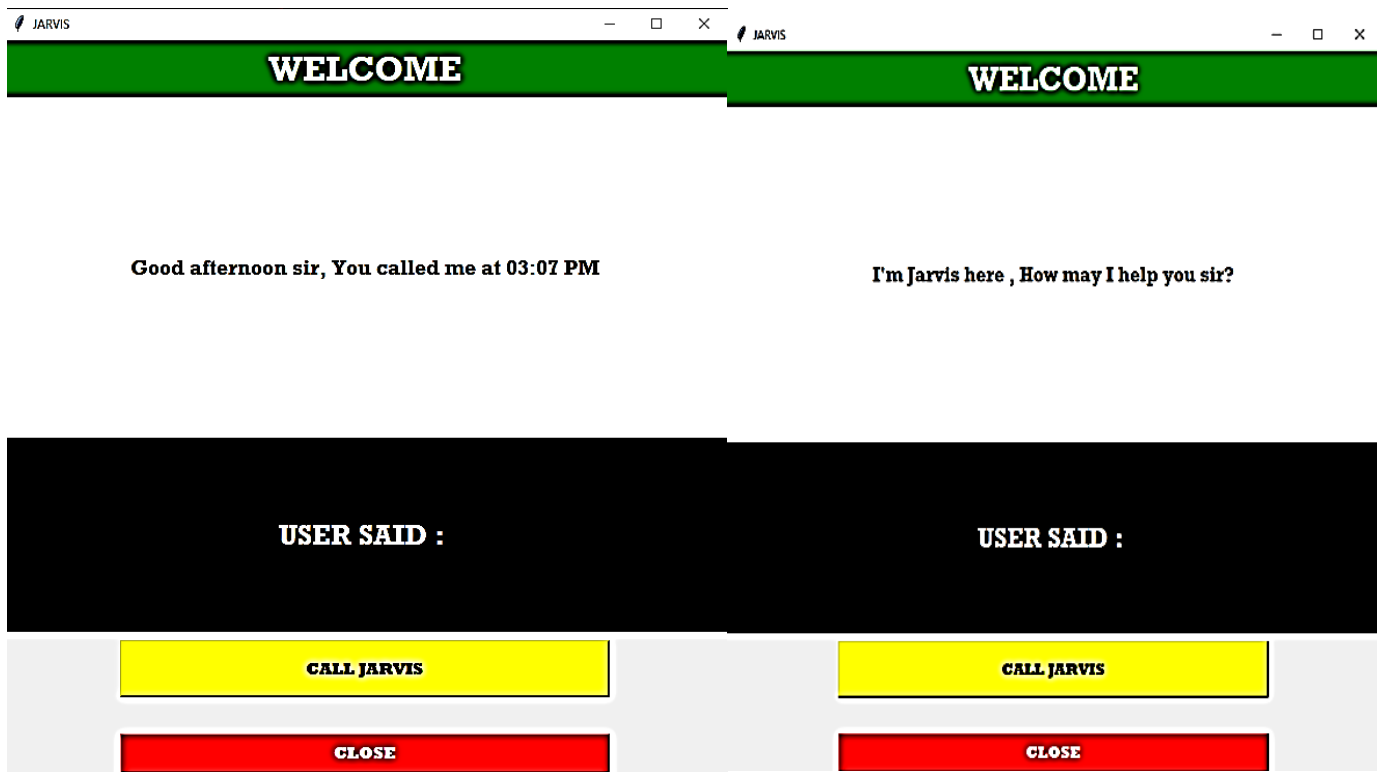
```
Run: ai.py x
C:\Users\Venkat\AppData\Local\Programs\Python\Pyth
JARVIS SAID : Initializing Jarvis
```

A screenshot of a Python terminal window. The title bar shows 'Run: ai.py x'. The command prompt is 'C:\Users\Venkat\AppData\Local\Programs\Python\Pyth'. The output is 'JARVIS SAID : Initializing Jarvis'.

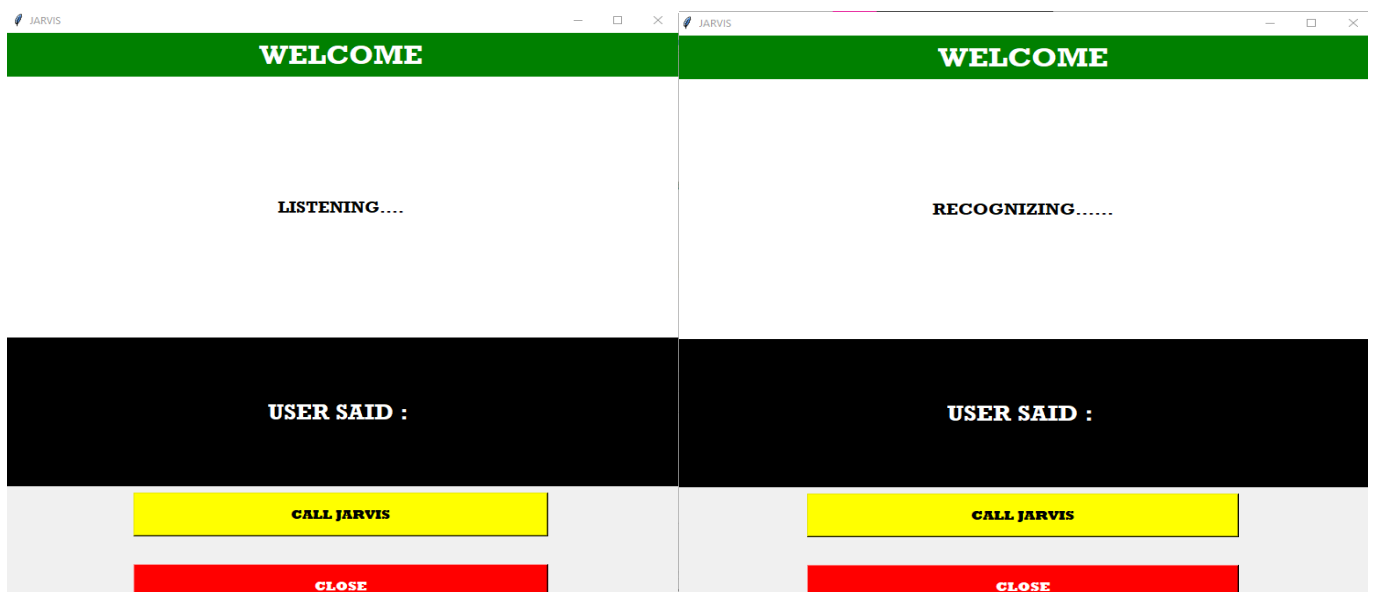
AFTER THE INITIALIZATION IT OPENS THE GUI



AFTER CALLING JARVIS



LISTENING AND RECOGNIZING

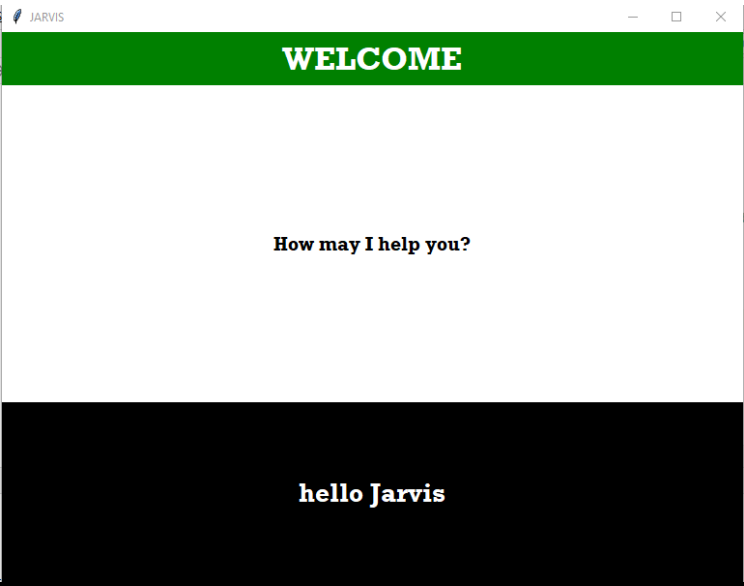
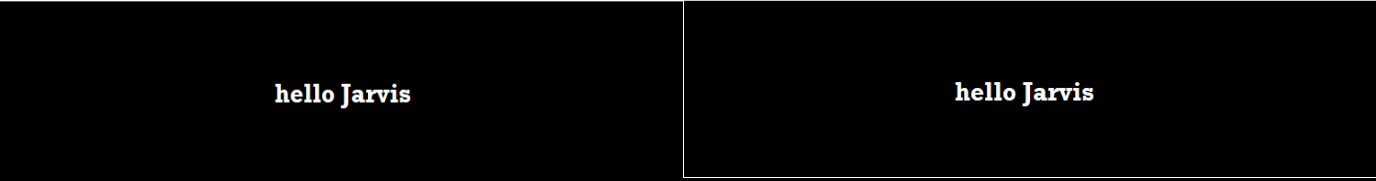


BASIC COMMUNICATION

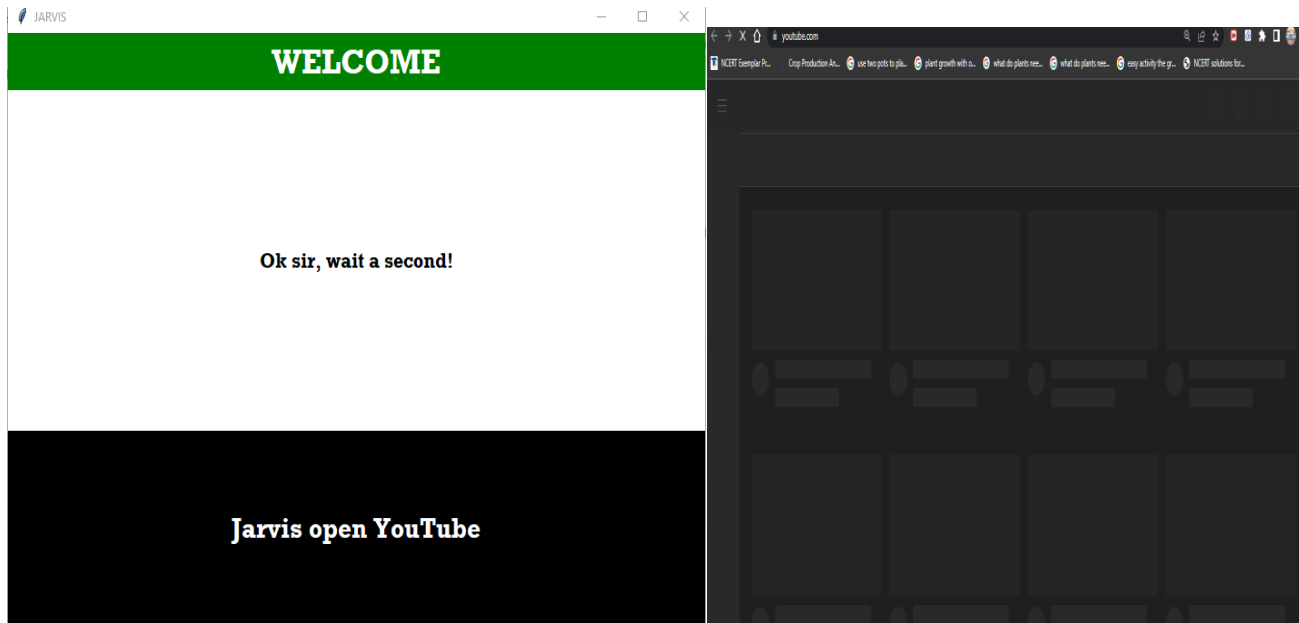


Hello sir, I am Jarvis

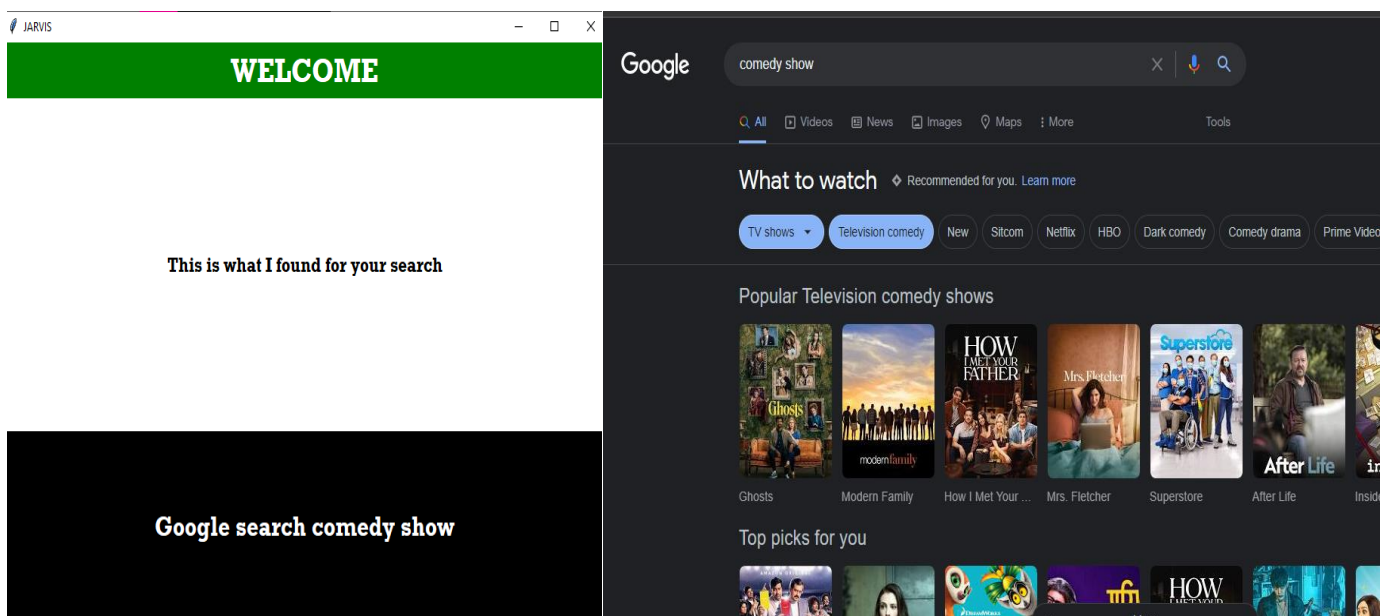
Your personal AI assistant!



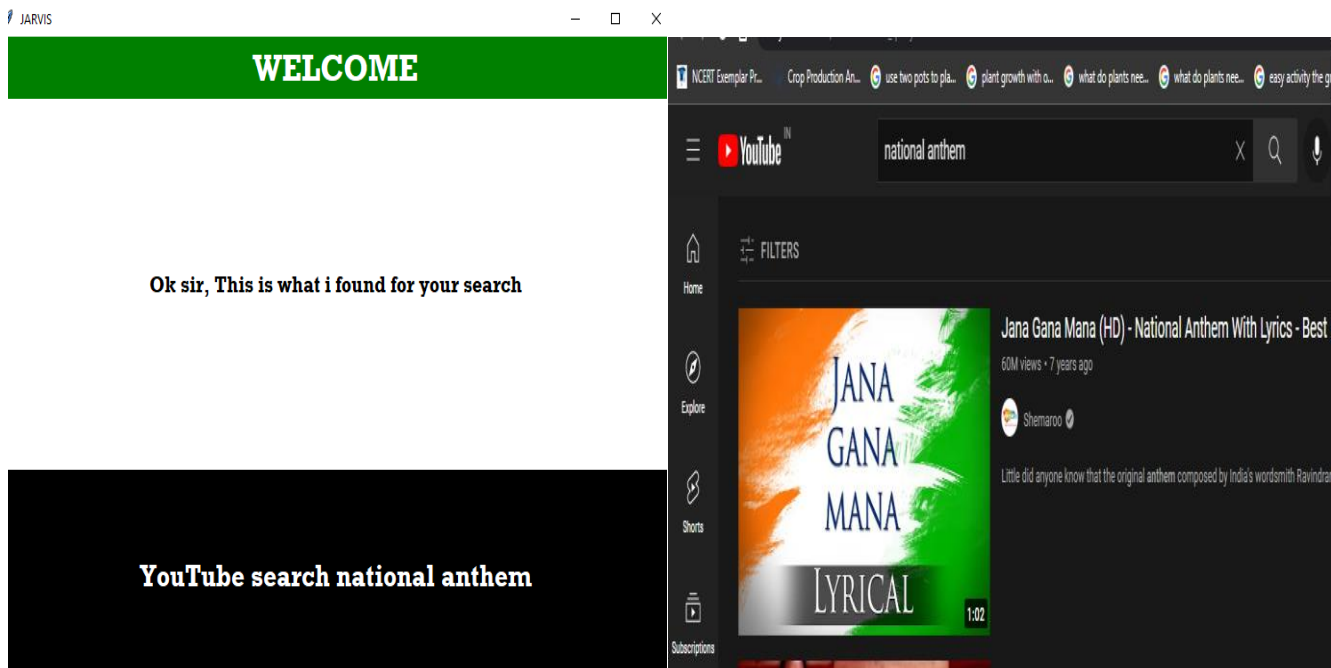
OPENING YOUTUBE USING JARVIS



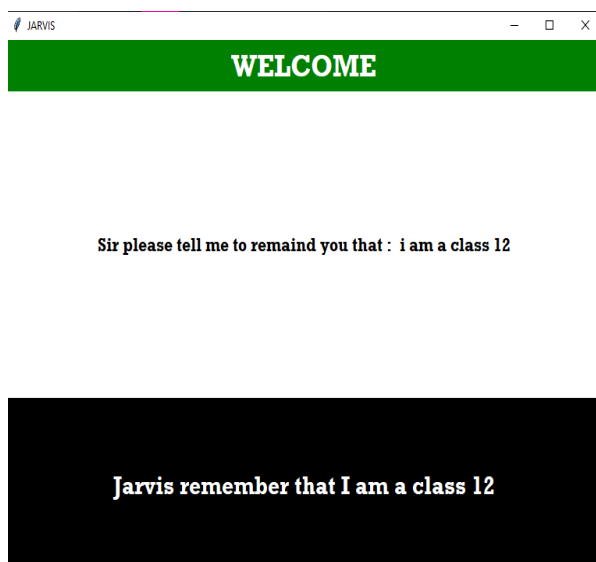
GOOGLE SEARCH



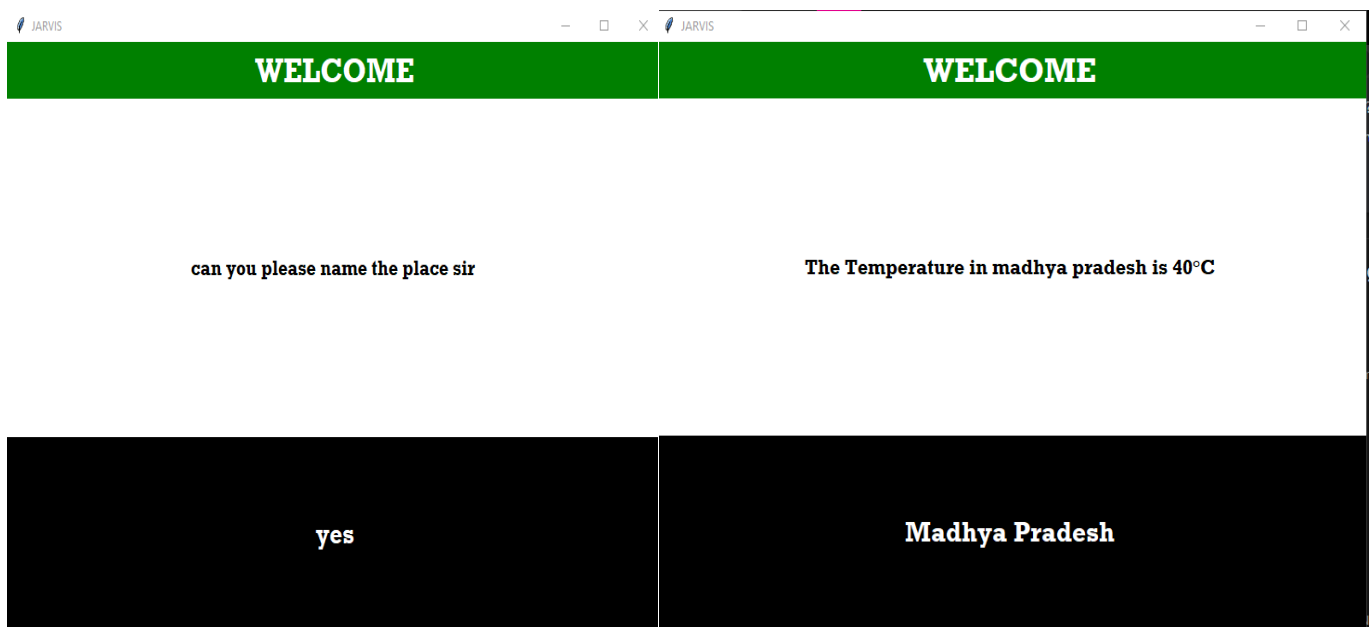
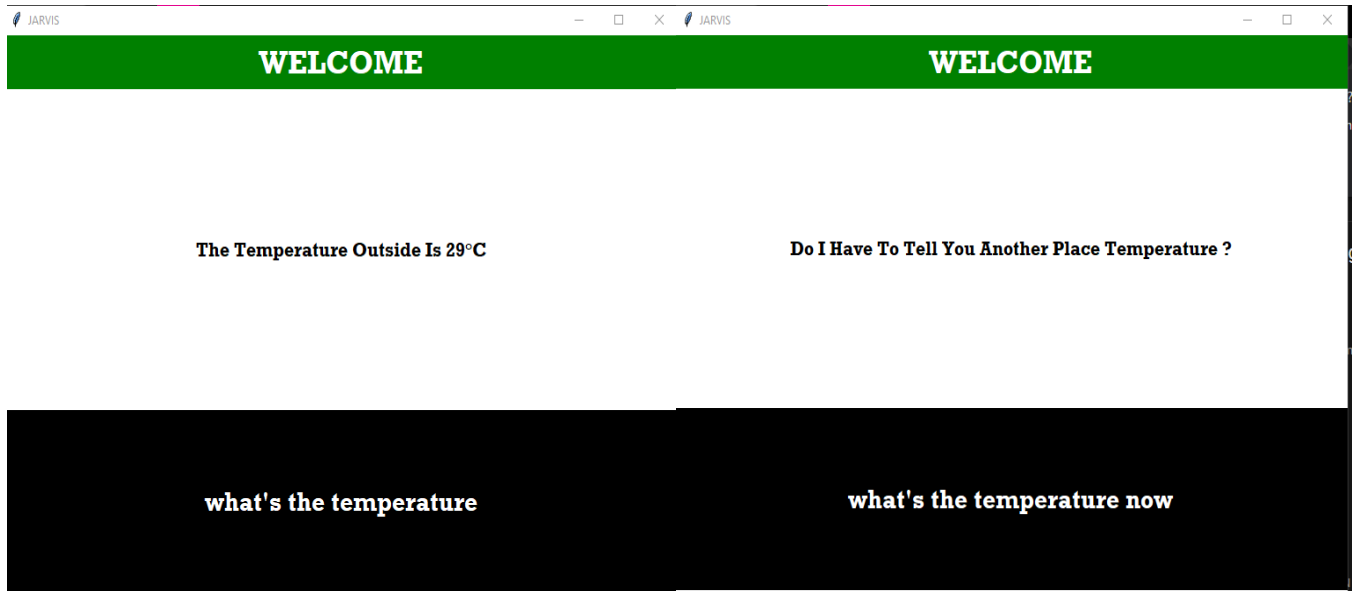
YOUTUBE SEARCH



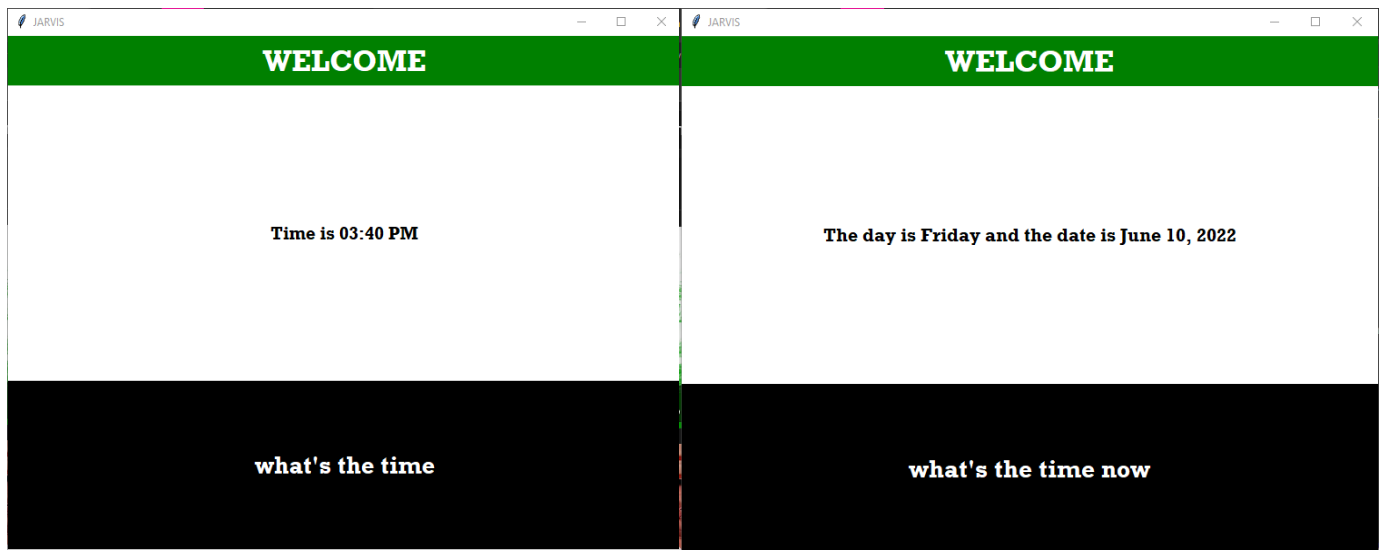
REMEMBER COMMAND



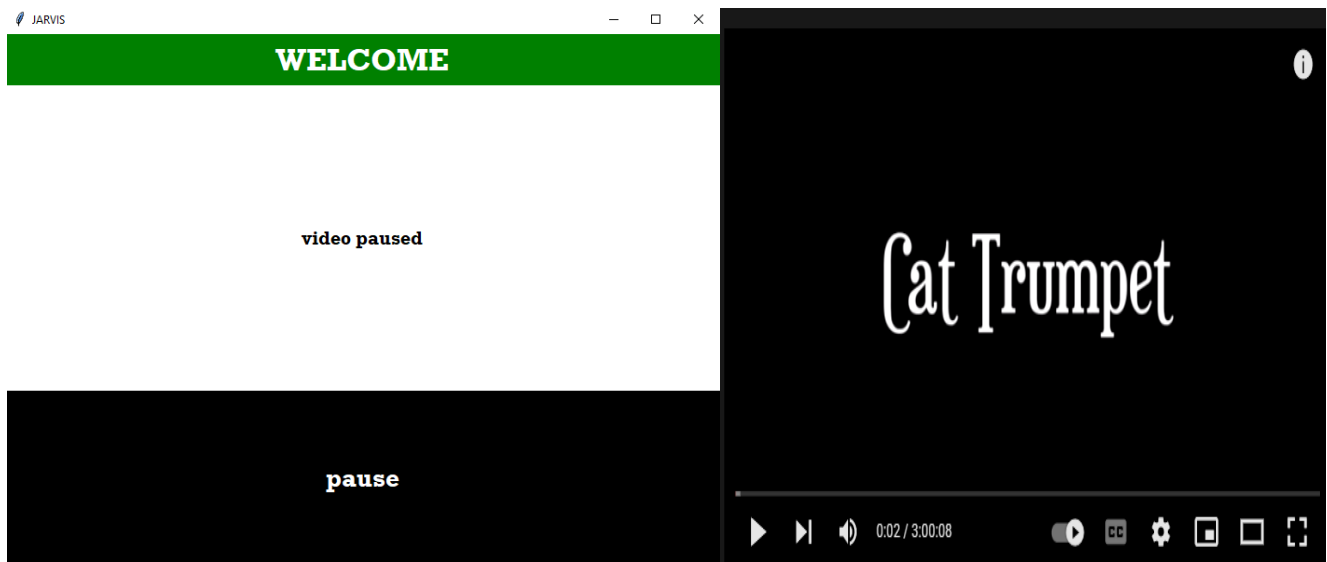
TEMPERATURE



TIME



PAUSING AND PLAYING THE VIDEO (AUTOMATION)



JARVIS

WELCOME

video played

play the video



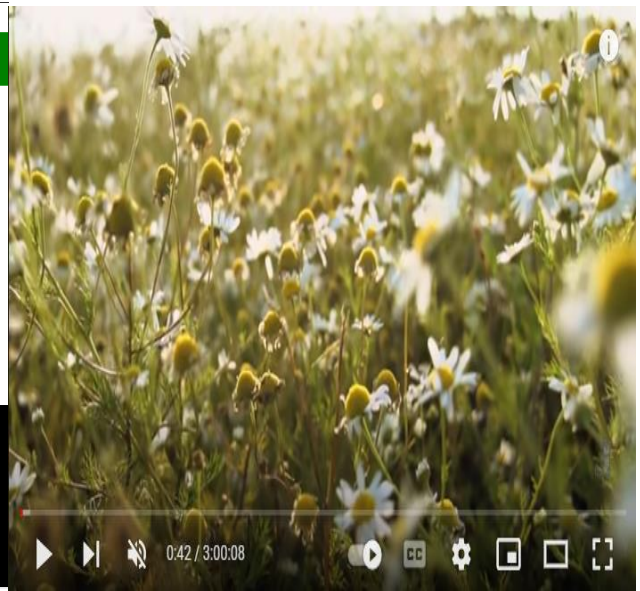
MUTE

JARVIS

WELCOME

video muted

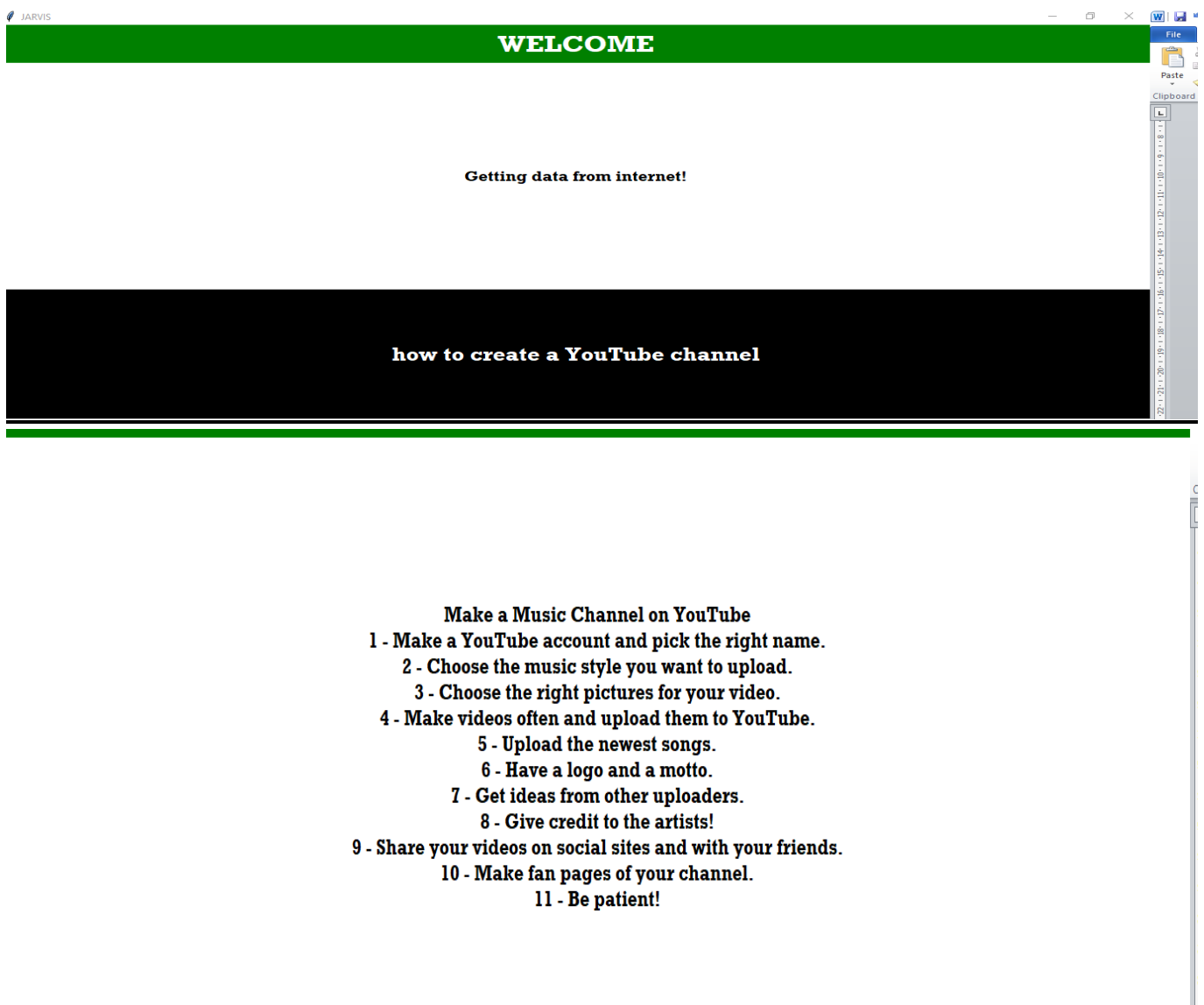
mute



JOKES COMMAND



COMMAND USING HOW TO -----



QUIT COMMAND FOR JARVIS

JARVIS

WELCOME

Going to sleep sir

Java sleep

JARVIS SAID : Going to sleep sir

Process finished with exit code 0

5. FUTURE SCOPE OF THIS PROJECT

DESIGN IMPROVEMENTS

No program has a perfect design without any flaws; it is the same here in this program. Even though the program is completed with all the primary functions implemented and work properly, there are still many things that can be done with this program. As the future improvement, the potential work that can be implemented ranging from adding more functions to offering the user a more comprehensive, convenient program, refining the logic to make the program more humanized and easy to use, increase the database capacity and add more possible keywords, responses and data in this program, interface optimization and etc.

ADDITIONAL FUNCTIONS

Add more functions: although there have been 15 normal functions that are used really often with the mobile phone, there can be more functions which simplify our daily life and make it convenient to use. Functions as playing movies, checking stocks, exchange rate, downloading and uploading, installing APPs and etc, these can be the potential functions that make the program more comprehensive and people can enjoy more services in this program.

DATABASE CAPACITY

Add database capacity and more humanized logical design; the program has a predefined logic to make it work with the corresponding commands. Thus, the user need to follow the structure of the commands, contain the dedicated keywords and well formalize the commands to work with each of the functions. In other words, the program is limited by the database capacity and no solution will be found if the user gives commands that are not readable by the program. Even if two commands have the same meaning and should get exactly same result set, the result might be that of one is working and the other one fails. Hence, the program is to some extent limited by the vocabulary and can be further optimized.

IMPROVED INTERFACE

Interface optimization, the interface can be further improved to make it nice to the users. Currently the interface design meets the basic requirement to present everything for this program, and the users are able to interact with the program through this interface, but the interface can always be optimized and more suitable constructed.

BIBLIOGRAPHY

<https://extrudesign.com/virtual-assistant-using-python/>

<https://www.geeksforgeeks.org/voice-assistant-using-python/>

<https://dev.to/mshrish/making-a-simple-voice-controlled-personal-assistant-interface-using-python-5ce1>

<https://sr9717943.medium.com/jarvis-desktop-assistant-with-gui-using-python-e95115ec0bc8>

<https://www.kdnuggets.com/2019/09/build-your-first-voice-assistant.html>

<https://www.instructables.com/How-to-Make-a-GUI-Virtual-Assistant/>

NOTES