Virtual AI Assistant

Submitted in partial fulfillment of the requirements
of the degree of

Master of Computer Application

by

Rupesh Bovalekar (01)
Ronak Pathare (42)
Siddhesh Waman (60)

Supervisor:
Mr. Ameya Parkar

V.E.S.
Since 1962

Department of MCA

Vivekanand Education Society's Institute of Technology
Master of Computer Application

Mumbai University

(2020-2021)

CERTIFICATE

This is to certify that the project entitled **“Virtual AI Assistant”** is a bonafide work of “**Rupesh Bovalekar (01), Ronak Pathare (42), Siddhesh Waman (60)”** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **“Postgraduate”** in **“Masters of Computer Application”**.

(Name and sign) Supervisor/Guide	(Name and sign) Co-Supervisor/Guide
(Name and sign) Head of Department	(Name and sign) Principal

ACKNOWLEDGEMENT

I would like to acknowledge our debt to each and every person associated in this project development. The project development required huge commitment from all the individuals involved in it.

I am thankful to principal **Dr. J. M. Nair** and Vice principal **Dr. M. Vijayalaxmi** of **Vivekanand Education Society's Institute of Technology (VESIT)** for providing the all facilities required for carrying out my research work.

It is a pleasure to express our deep and sincere gratitude to project guide **Mr. Ameya Parkar** and the profoundly grateful towards the unmatched help rendered by her.

We would like to thank our Head of the department **Dr. Shivkumar Goel** for his blessings and for being a constant source of inspiration to us. We are also thankful to all the professors for making their valued contribution and for providing a helping hand to us in times of queries and problems.

I would also thank my friends for giving me the opinions and various inputs in long discussion on the project which helped me shape the project keeping in mind the user friendly.

I would also like to thank everyone helped me in my project in some way or other which includes providing me with some information.

Thanking you.

RUPESH BOVALEKAR

RONAK PATHARE

SIDDHESH WAMAN

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

RUPESH BOVALEKAR

RONAK PATHARE

SIDDHESH WAMAN

Date : 01 - 09 - 2021

TABLE OF CONTENTS

INTRODUCTION

In today’s era almost all tasks are digitalized. We have Smartphone in hands and it is nothing less than having world at your fingertips. These days we aren’t even using fingers. We just speak of the task and it is done. There exist systems where we can say Text Dad, “I’ll be late today.” And the text is sent. That is the task of a Virtual Assistant. It also supports specialized task such as booking a flight, or finding cheapest book online from various e-commerce sites and then providing an interface to book an order are helping automate search, discovery and online order operations.

Virtual Assistants are software programs that help you ease your day to day tasks, such as showing weather report, creating reminders, making shopping lists etc. They can take commands via text (online chat bots) or by voice. Voice based intelligent assistants need an invoking word or wake word to activate the listener, followed by the command. For my project the wake word is AI Assistant . We have so many virtual assistants, such as Apple’s Siri, Amazon’s Alexa and Microsoft’s Cortana. For this project, wake word was chosen AI Assistant.

This system is designed to be used efficiently on desktops. Personal assistant software improves user productivity by managing routine tasks of the user and by providing information from online sources to the user. AI Assistant is effortless to use. Call the wake word ‘AI Assistant’ followed by the command and within seconds, it gets executed.

Voice searches have dominated over text search. Web searches conducted via mobile devices have only just overtaken those carried out using a computer and the analysts are already predicting that 50% of searches will be via voice by 2020.Virtual assistants are turningout to be smarter than ever. Allow your intelligent assistant to make email work for you. Detect intent, pick out important information, automate processes, and deliver personalized responses.

This project was started on the premise that there is 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.

BACKGROUND

There already exist a number of desktop virtual assistants. A few examples of current virtual assistants available in market are discussed in this section along with the tasks they can provide and their drawbacks.

- **SIRI from Apple**

SIRI is personal assistant software that interfaces with the user thru voice interface, recognizes commands and acts on them. It learns to adapt to user’s speech and thus improves voice recognition over time. It also tries to converse with the user when it does not identify the user request.

It integrates with calendar, contacts and music library applications on the device and also integrates with GPS and camera on the device. It uses location, temporal, social and task based contexts, to personalize the agent behavior specifically to the user at a given point of time.

Supported Tasks

- Call someone from my contacts list
- Launch an application on my iPhone
- Send a text message to someone
- Set up a meeting on my calendar for 9am tomorrow
- Set an alarm for 5am tomorrow morning
- Play a specific song in my iTunes library
- Enter a new note

Drawback

SIRI does not maintain a knowledge database of its own and its understanding comes from the information captured in domain models and data models.

- **ReQall**

ReQall is personal assistant software that runs on smartphones running Apple iOS or Google Android operating system. It helps user to recall notes as well as tasks within a location and time context. It records user inputs and converts them into commands, and monitors current stack of user tasks to proactively suggest actions while considering any changes in the environment. It also presents information based on the context of the user, as well as filter information to the user based on its learned understanding of the priority of that information.

Supported Tasks

- Reminders
- Email
- Calendar, Google Calendar
- Outlook
- Evernote
- Facebook, LinkedIn
- News Feeds

Drawback

Will take some time to put all of the to-do items in – you could spend more timeputting the entries in than actually doing the revision.

OBJECTIVES

Main objective of building personal assistant software (a virtual assistant) is using semantic data sources available on the web, user generated content and providing knowledge from knowledge databases. The main purpose of an intelligent virtual assistant is to answer questions that users may have. This may be done in a business environment, for example, on the business website, with a chat interface. On the mobile platform, the intelligent virtual assistant is available as a call-button operated service where a voice asks the user “What can I do for you?” and then responds to verbal input.

Virtual assistants can tremendously save you time. We spend hours in online research and then making the report in our terms of understanding. AI Assistant can do that for you. Provide a topic for research and continue with your tasks while AI Assistant does the research. Another difficult task is to remember test dates, birthdates or anniversaries. It comes with a surprise when you enter the class and realize it is class test today. Just tell AI Assistant in advance about your tests andshe reminds you well in advance so you can prepare for the test.

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: whereas we can write about 40 words per minute, we are capable of speaking around 150 during the same period of time15. In this respect, the ability of personal assistants to accurately recognize spoken words is a prerequisite for them to be adopted by consumers.

PURPOSE, SCOPE AND APPLICABILITY

• Purpose

Purpose of virtual assistant is to being capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news. Virtual assistants enable users to speak natural language voice commands in order to operate the device and its apps.

There is an increased overall awareness and a higher level of comfort demonstrated specifically by millennial consumers. In this ever-evolving digital world where speed, efficiency, and convenience are constantly being optimized, it’s clear that we are moving towards less screen interaction.

• Scope

Voice assistants will continue to offer more *individualized* experiences as they get better at differentiating between voices. However, it’s not just developers that need to address the complexity of developing for voice as brands also need to understand the capabilities of each device and integration and if it makes sense for their specific brand. They will also need to focus on maintaining a user experience that is consistent within the coming years as complexity becomes more of a concern. This is because the visual interface with voice assistants is missing. Users simply cannot see or touch a voice interface.

• Applicability

The mass adoption of artificial intelligence in users’ everyday lives is also fueling the shift towards voice. The number of IoT devices such as smart thermostats and speakers are giving voice assistants more utility in a connected user’s life. Smart speakers are the number one way we are seeing voice being used. Many industry experts even predict that nearly every application will integrate voice technology in some way in the next 5 years.

The use of virtual assistants can also enhance the system of IoT (Internet of Things). Twenty years from now, Microsoft and its competitors will be offering personal digitalassistants that will offer the services of a full-time employee usually reserved for the rich and famous.

SURVEY OF TECHNOLOGY

- **Python**

Python is an OOPs (Object Oriented Programming) based, high level, interpreted programming language. It is a robust, highly useful language focused on rapid application development (RAD). Python helps in easy writing and execution of codes. Python can implement the same logic with as much as 1/5th code as compared to other OOPslanguages.

Python provides a huge list of benefits to all. The usage of Python is such that it cannot be limited to only one activity. Its growing popularity has allowed it to enter into some of the most popular and complex processes like Artificial Intelligence (AI), Machine Learning (ML), natural language processing, data science etc. Python has a lot of libraries for every need of this project. For AI Assistant , libraries used are speechrecognition to recognize voice, Pyttsx for text to speech, selenium for web automation etc.

Python is reasonably efficient. Efficiency is usually not a problem for small examples. If your Python code is not efficient enough, a general procedure to improve it is to find out what is taking most the time, and implement just that part more efficiently in some lower-level language. This will result in much less programming and more efficient code (because you will have more time to optimize) than writing everything in a low-level language.

REQUIREMENT AND ANALYSIS

System Analysis is about complete understanding of existing systems and finding where the existing system fails. The solution is determined to resolve issues in the proposed system. It defines the system. The system is divided into smaller parts. Their functions and inter relation of these modules are studied in system analysis. The complete analysis is followed below.

PROBLEM DEFINITION

Usually, user needs to manually manage multiple sets of applications to complete one task. For example, a user trying to make a travel plan needs to check for airport codes for nearby airports and then check travel sites for tickets between combinations of airports to reach the destination. There is need of a system that can manage tasks effortlessly.

We already have multiple virtual assistants. But we hardly use it. There are number of people who have issues in voice recognition. These systems can understand English phrases but they fail to recognize in our accent. Our way of pronunciation is way distinct from theirs. Also, they are easy to use on mobile devices than desktop systems. There is need of a virtual assistant that can understand English in Indian accent and work on desktop system.

When a virtual assistant is not able to answer questions accurately, it’s because it lacks the proper context or doesn’t understand the intent of the question. Its ability to answer questions relevantly only happens with rigorous optimization, involving both humans and machine learning. Continuously ensuring solid quality control strategies will also help managethe risk of the virtual assistant learning undesired bad behaviors. They require large amount of information to be fed in order for it to work efficiently.

Virtual assistant should be able to model complex task dependencies and use these models to recommend optimized plans for the user. It needs to be tested for finding optimum paths when a task has multiple sub-tasks and each sub-task can have its own sub-tasks. In such a case there can be multiple solutions to paths, and the it should be able to consider user preferences, other active tasks, priorities in order to recommend a particular plan.

REQUIREMENT SPECIFICATION

Personal 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.

AI Assistant focuses on relieving the user of entering text input and using voice as primary means of user input. Agent then applies voice recognition algorithms to this input and records the input. It then use this input to call one of the personal information management applications such as task list or calendar to record a new entry or to search about it on search engines like Google, Bing or Yahoo etc. Focus is on capturing the user input through voice, recognizing the input and then executing the tasks if the agent understands the task. Software takes this input in natural language, and so makes it easier for the user to input what he or she desires to be done.

Voice recognition software enables hands free use of the applications, lets users to query or command the agent through voice interface. This helps users to have access to the agent while performing other tasks and thus enhances value of the system itself. AI Assistant also have ubiquitous connectivity through Wi-Fi or LAN connection, enabling distributed applications that can leverage other APIs exposed on the web without a need to store them locally.

Virtual assistants must provide a wide variety of services. These include:

- Providing information such as weather, facts from e.g. Wikipedia etc.
- Set an alarm or make to-do lists and shopping lists.
- Remind you of birthdays and meetings.
- Play music from streaming services such as Saavn and Gaana.
- Play videos, TV shows or movies on televisions, streaming from e.g. Netflix or Hotstar.
- Book tickets for shows, travel and movies.

FEASIBILITY STUDY

Feasibility study can help you determine whether or not you should proceed with your project. It is essential to evaluate cost and benefit. It is essential to evaluate cost and benefit of the proposed system. Five types of feasibility study are taken into consideration.

1. **Technical feasibility:** It includes finding out technologies for the project, both hardware and software. For virtual assistant, user must have microphone to convey their message and a speaker to listen when system speaks. These are very cheap now a days and everyone generally possess them. Besides, system needs internet connection. While using AI Assistant , make sure you have a steady internet connection. It is also not an issue in this era where almost every home or office has Wi-Fi.
2. **Operational feasibility:** It is the ease and simplicity of operation of proposed system. System does not require any special skill set for users to operate it. In fact, it is designed to be used by almost everyone. Kids who still don't know to write can read out problems for system and get answers.
3. **Economical feasibility:** Here, we find the total cost and benefit of the proposed system over current system. For this project, the main cost is documentation cost. User also would have to pay for microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, AI Assistant won't cost too much.
4. **Organizational feasibility:** This shows the management and organizational structure of the project. This project is not built by a team. The management tasks are all to be carried out by a single person. That won't create any management issues and will increase the feasibility of the project.
5. **Cultural feasibility:** It deals with compatibility of the project with cultural environment. Virtual assistant is built in accordance with the general culture. The project is named AI Assistant so as to represent Indian culture without undermining local beliefs.

This project is technically feasible with no external hardware requirements. Also it is simple in operation and does not cost training or repairs. Overall feasibility study of the project reveals that the goals of the proposed system are achievable. Decision is taken to proceed with the project.

SOFTWARE AND HARDWARE REQUIREMENTS

The software is designed to be light-weighted so that it doesn't be a burden on the machine running it. This system is being build keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirement for virtual assistant.

Hardware:

- Pentium-pro processor or later.
- RAM 512MB or more.

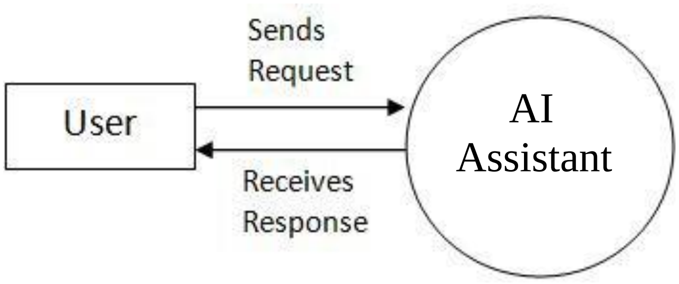
Software:

- Windows 7(32-bit) or above.
- Python 2.7 or later
- Chrome Driver
- Selenium Web Automation
- SQLite

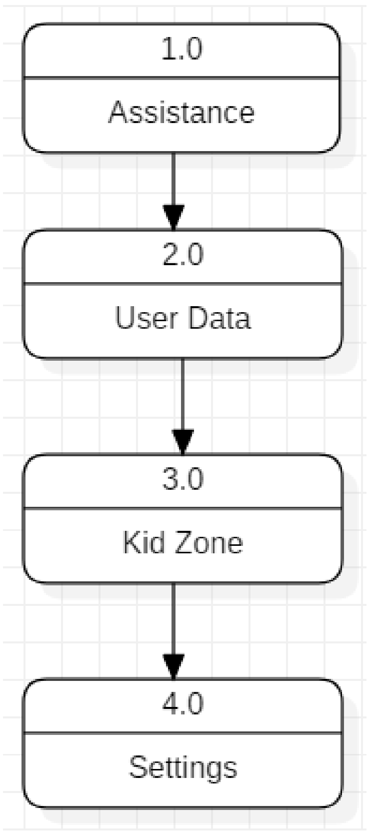
SYSTEM DESIGN

Data Flow Diagram

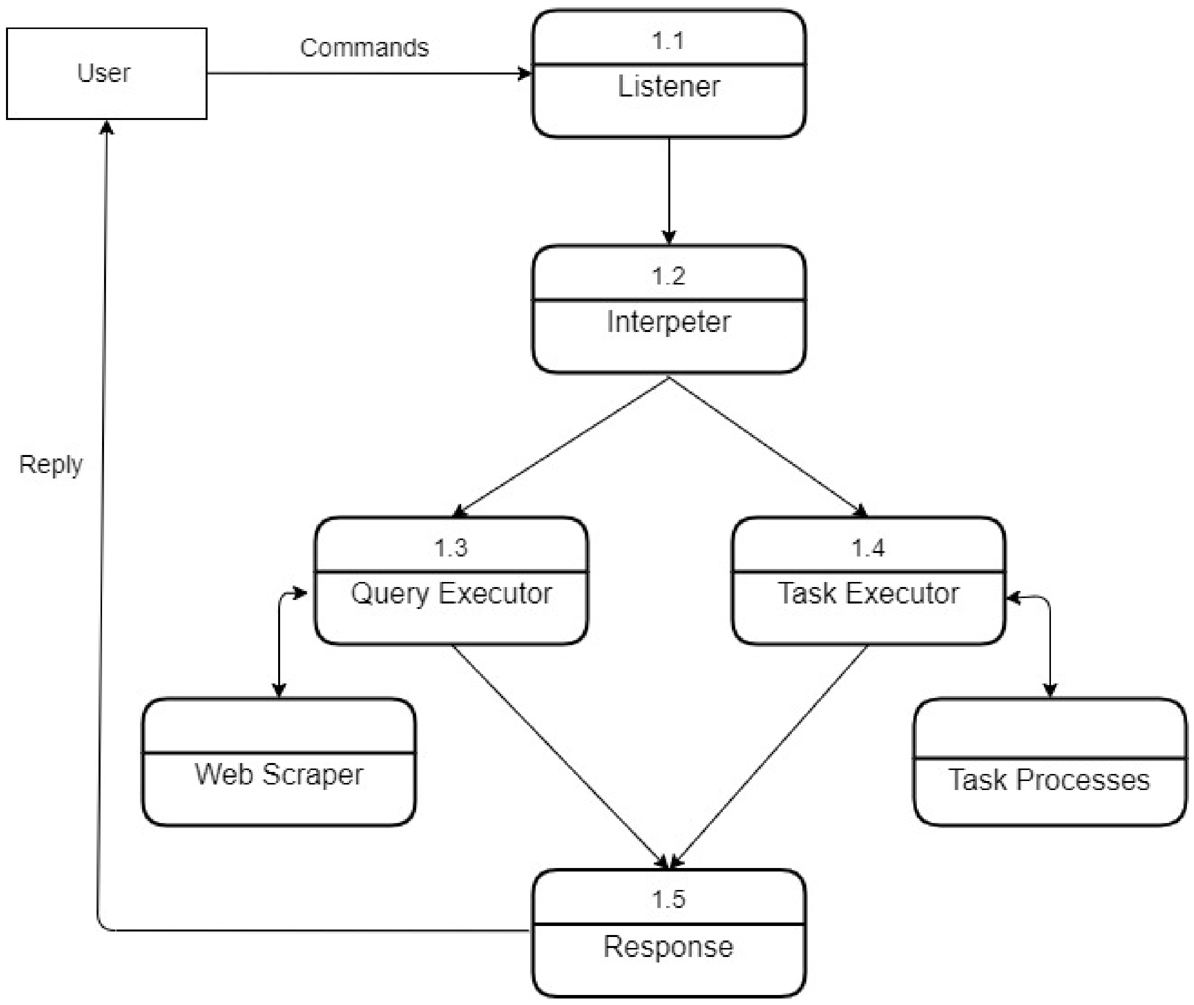
- DFD Level 0 (Context Level Diagram)



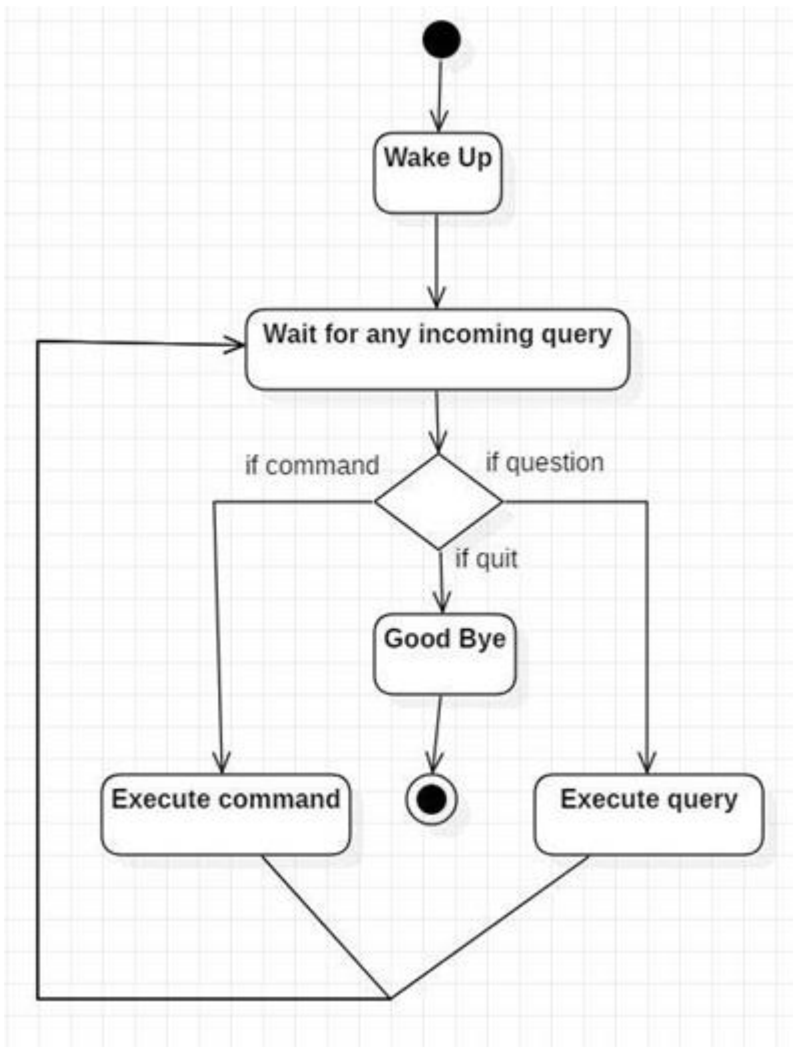
- DFD Level 1



- DFD Level 2

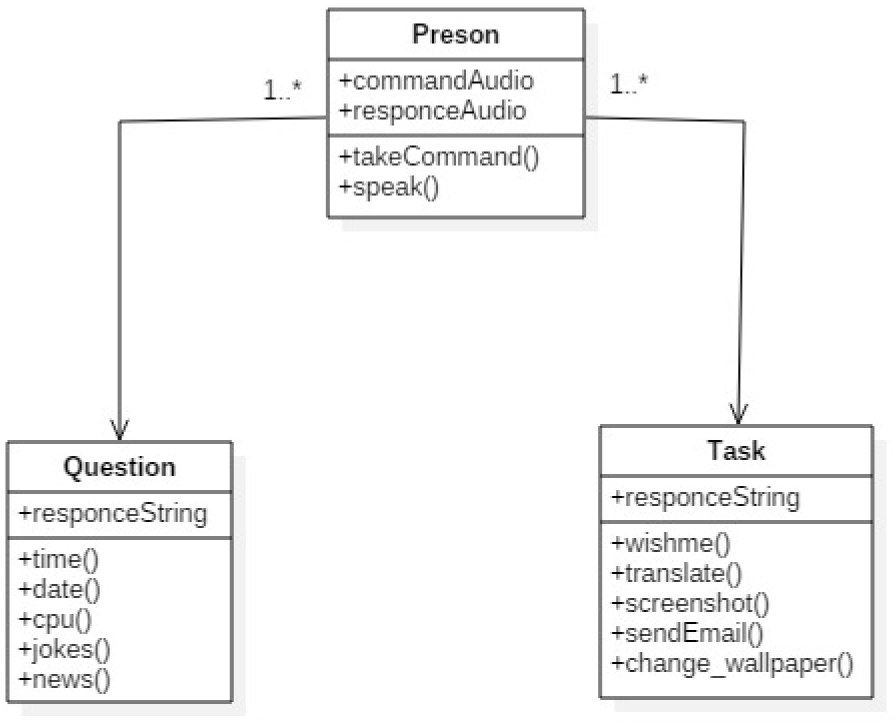

User
Commands
1.1
Listener
1.2
Interpeter
Reply
1.3
Query Executor
1.4
Task Executor
Web Scraper
Task Processes
1.5
Response

Activity Diagram



Initially, the system is in idle mode. As it receives any wakeup call it begins execution. The received command is identified whether it is a questionnaire or a task to be performed. Specific action is taken accordingly. After the Question is being answered or the task is beingperformed, the system waits for another command. This loop continues unless it receives quitcommand. At that moment, it goes back to sleep.

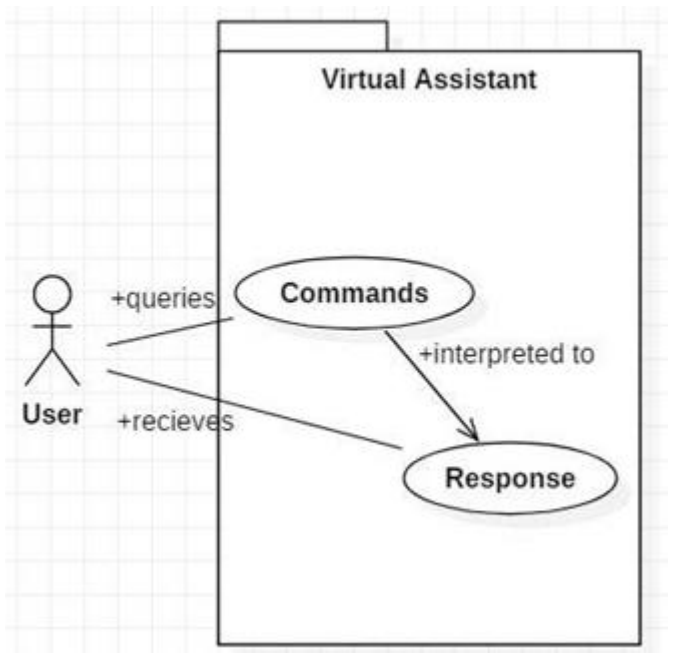
Class Diagram



The class user has 2 attributes command that it sends in audio and the response it receives which is also audio. It performs function to listen the user command. Interpret it and then reply or sends back response accordingly. Question class has the command in string form as it is interpreted by interpret class. It sends it to general or about or search function based on its identification.

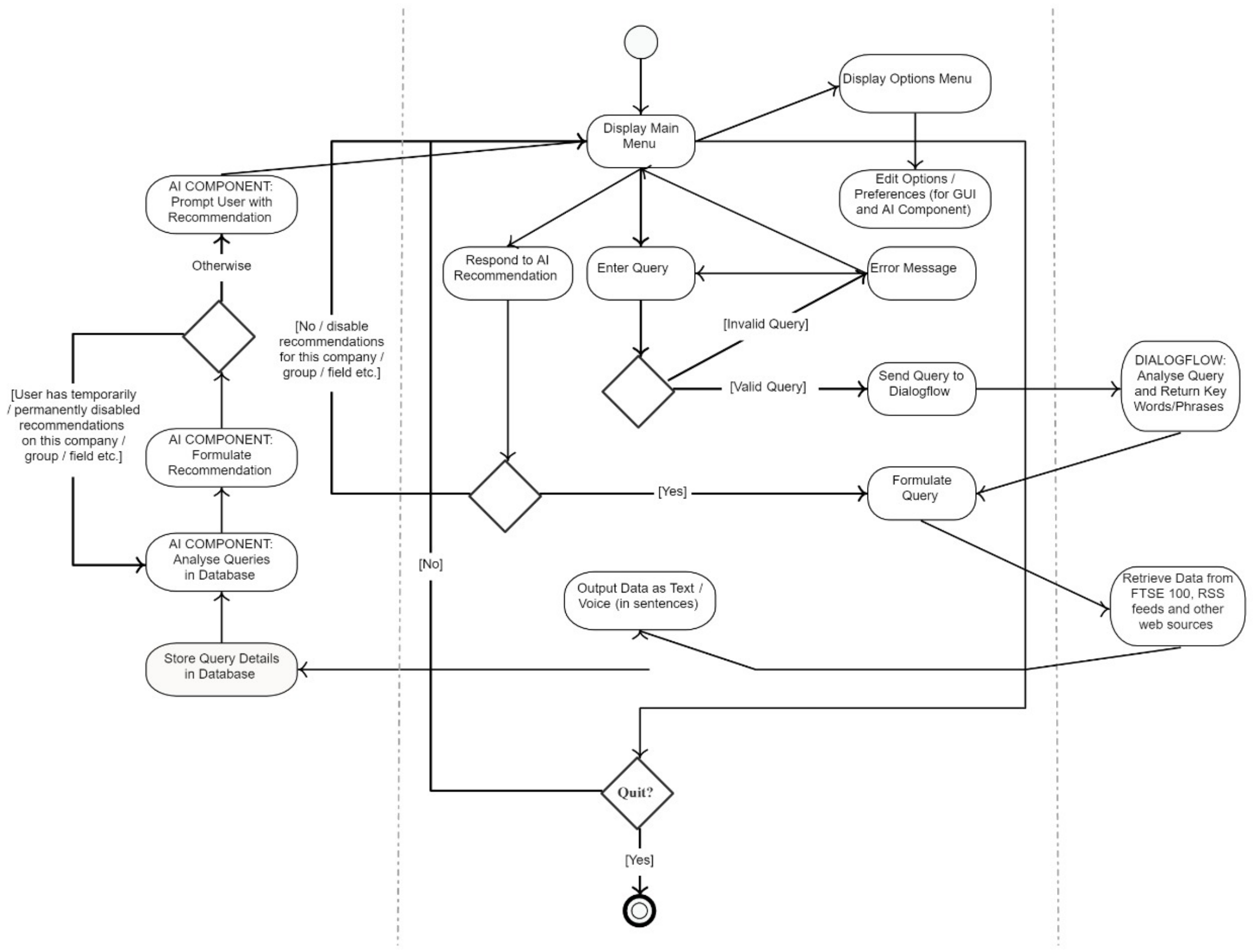
The task class also has interpreted command in string format. It has various functions like reminder, note, mimic, research and reader.

Use Case Diagram



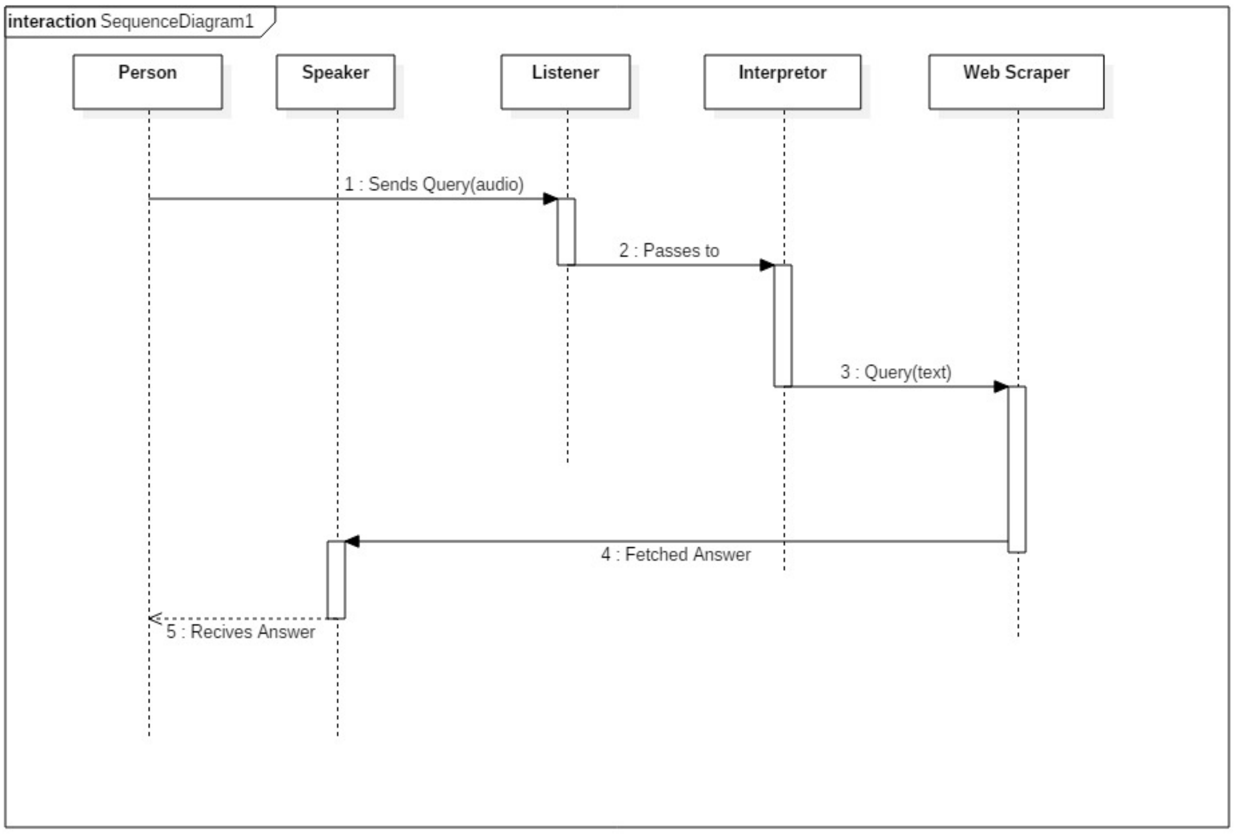
In this project there is only one user. The user queries command to the system. System then interprets it and fetches answer. The response is sent back to the user.

State Diagram


Display Main Menu
Display Options Menu
Edit Options / Preferences (for GUI and AI Component)
AI COMPONENT: Prompt User with Recommendation
Otherwise
Respond to AI Recommendation
Enter Query
Error Message
[Invalid Query]
[No / disable recommendations for this company / group / field etc.]
[Valid Query]
Send Query to Dialogflow
DIALOGFLOW: Analyse Query and Return Key Words/Phrases
[User has temporarily / permanently disabled recommendations on this company / group / field etc.]
AI COMPONENT: Formulate Recommendation
[Yes]
Formulate Query
AI COMPONENT: Analyse Queries in Database
[No]
Output Data as Text / Voice (in sentences)
Retrieve Data from FTSE 100, RSS feeds and other web sources
Store Query Details in Database
Quit?
[Yes]

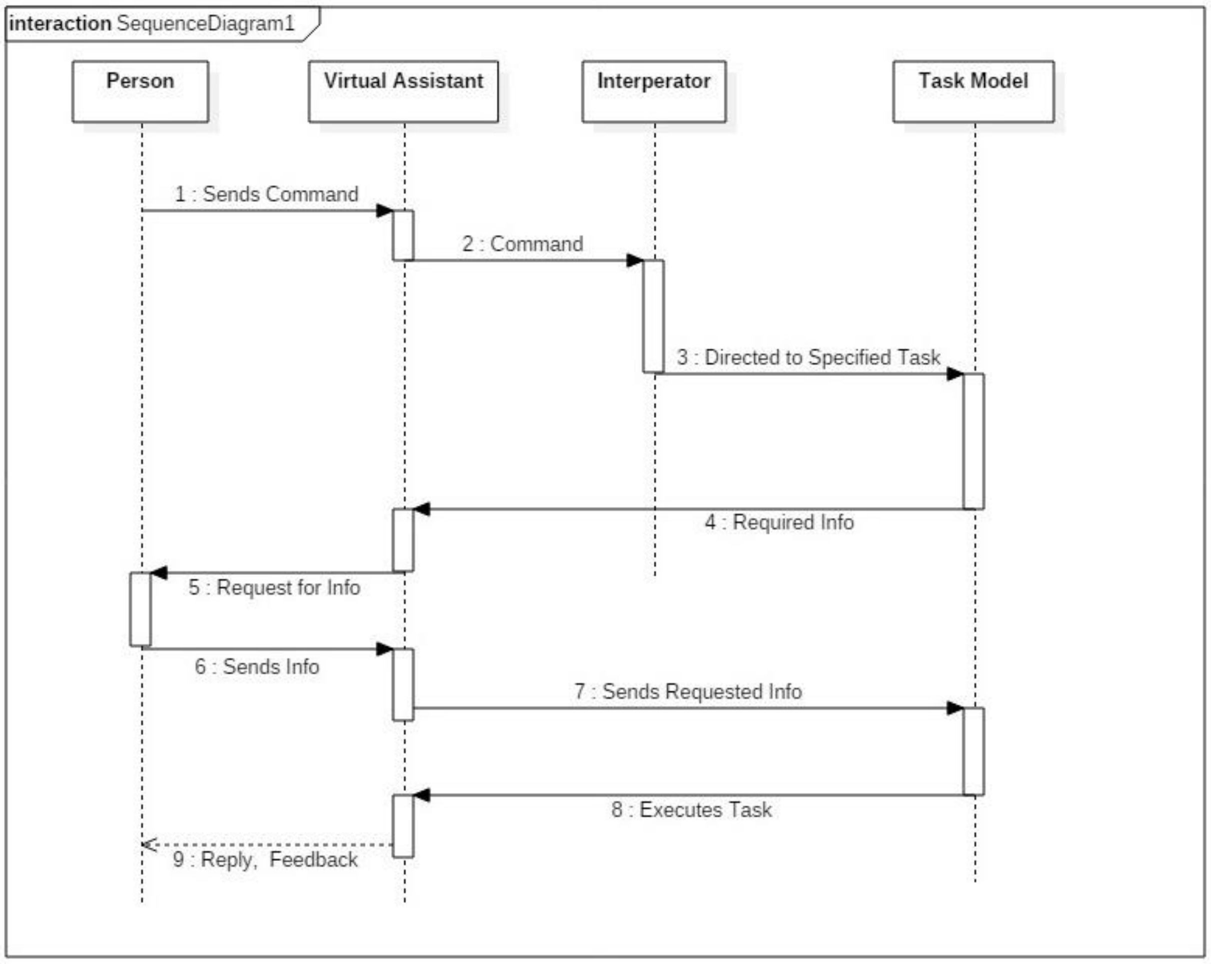
Sequence Diagram

Sequence diagram for Query-Response



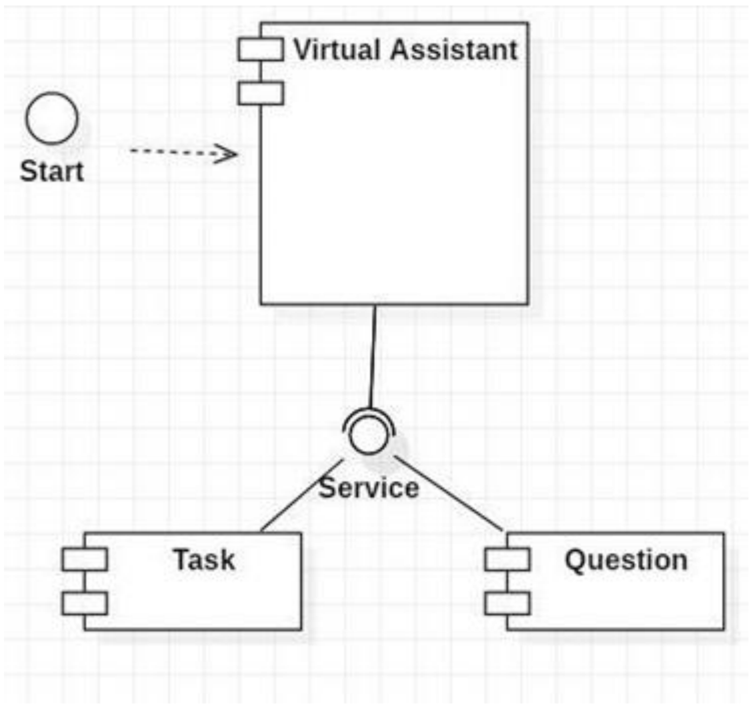
The above sequence diagram shows how an answer asked by the user is being fetched from internet. The audio query is interpreted and sent to Web scraper. The web scraper searches and finds the answer. It is then sent back to speaker, where it speaks the answer to user.

Sequence Diagram For Task Execution



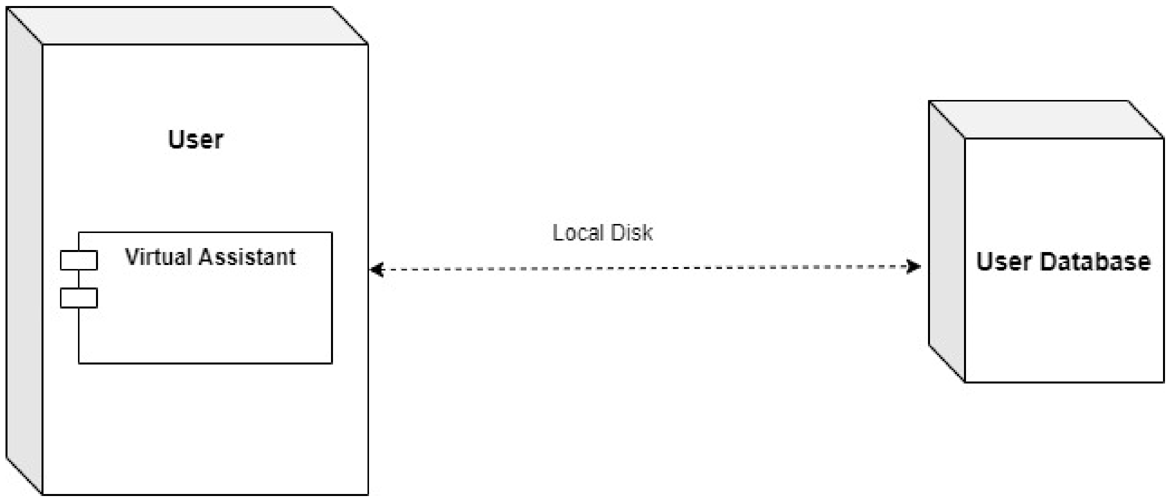
The user sends command to virtual assistant in audio form. The command is passed to the interpreter. It identifies what the user has asked and directs it to task executer. If the task ismissing some info, the virtual assistant asks user back about it. The received information is sent back to task and it is accomplished. After execution feedback is sent back to user.

Component Diagram



The main component here is the Virtual Assistant. It provides two specific service,executing Task or Answering your question.

Deployment Diagram



The user interacts with SQLite database using SQLite connection in Python code. The knowledge database DBPedia must be accessed via internet connection. This requires LAN orWLAN / Ethernet network.

TEST CASE DESIGN

- **Test Case 1**

Test Title :- Response Time

Test ID :- T1

Test Priority :- High

Test Objective :- To make sure that the system respond back time is efficient.

Description :-

Time is very critical in a voice based system. As we are not typing inputs, we are speaking them. The system must also reply in a moment. User must get instant response of thequery made.

- **Test Case 2**

Test Title :- Accuracy

Test ID :- T2

Test Priority :- High

Test Objective :- To assure that answers retrieved by system are accurate as per gathered data.

Description :-

A virtual assistant system is mainly used to get precise answers to any question asked. Getting answer in a moment is of no use if the answer is not correct. Accuracy is of utmost importance in a virtual assistant system.

- **Test Case 3**

Test Title :- Approximation

Test ID :- T3

Test priority :- Moderate

Test Objective :- To check approximate answers about calculations.

Description :-

There are times when mathematical calculation requires approximate value. For example, if someone asks for value of PI the system must respond with approximate value andnot the accurate value. Getting exact value in such cases is undesirable.

- **Test Case 4**

Test Title :- Face recognition

Test ID :- T4

Test Priority :- High

Test Objective :- To make sure that the system respond back with user name by identifying user using computer via web camera or integrated camera of laptop

Description :-

This module enables our AI Assistant to identify current user using the computer by identifying user by capturing his image form web camera or integrated camera of laptop.

Note :- There might include a few more test cases and these test cases are also subject to changewith the final software development.

SYSTEM IMPLEMENTATION

Roma.py

```python
import pyttsx3
import datetime
import speech_recognition as sr
import pyaudio
import wikipedia
import smtplib
import webbrowser as wb
import os
import pyautogui
import psutil
import pyjokes
import selenium
import wolframalpha
from googletrans import Translator
from newsapi import NewsApiClient
from time import sleep
import unsplash_search
import requests
import wget
import subprocess
import ctypes
import getpass
import pywhatkit
import winsound
import threading
from pywikihow import search_wikihow
from faceRecognition_master import livefacerecognizer
from PyQt5 import QtWidgets,QtCore,QtGui, sip
from PyQt5.QtCore import QTimer,QTime, QDate,Qt
from PyQt5.QtGui import QMovie
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
from PyQt5.QtGui import *
from PyQt5.uic import loadUiType
from jarvisGUI import Ui_MainWindow
import sys

SPI_SETDESKWALLPAPER = 20
engine = pyttsx3.init()
client =wolframalpha.Client('5GWXGW-4KREWY9PW2')

def takeCommand():
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening.....")
        r.pause_threshold = 1
        audio = r.listen(source)
    try:
        print("Recognizing....")
        query =r.recognize_google(audio, language ='en-in')
        print(query)
    except Exception as e:
        print(e)
        speak("")
        return "None"
    return query

def speak(audio):
    engine.say(audio)
    engine.runAndWait()
def time():
    Time = datetime.datetime.now().strftime("%I:%M:%S")
    speak("The current time is")
    speak(Time)
def date():
    year = int(datetime.datetime.now().year)
    month = int(datetime.datetime.now().month)
    date = int(datetime.datetime.now().day)
    speak("The current date is")
    speak(date)
    speak(month)
    speak(year)
def wishme():
    speak("Welcome sir! I am Kalki Your Virtual AI Assistant")
    hour = datetime.datetime.now().hour
    if hour >=6 and hour <12:
        speak("Good Morning")
    elif hour >=12 and hour <18:
        speak("Good Afternoon")
    elif hour >=18 and hour <23:
        speak("Good Evening")
    else:
        speak("Good Night")
    speak("How can I help You")
def screenshot():
    img = pyautogui.screenshot()
    date =datetime.datetime.now().strftime("%m%d%Y_%H%M%S")
    path="C:\\Jarvis\\Screenshot\\ScreenShot_"+date+".png"
    filename = os.listdir("C:\\")
    if 'Jarvis' in filename:
        files= os.listdir("C:\Jarvis")
        if 'Screenshot' in files:
            img.save(path)
        else:
            os.mkdir("C:\\Jarvis\\Screenshot")
            img.save(path)
    else :
        os.makedirs("C:\\Jarvis\\Screenshot")
        img.save(path)
def cpu():
    usage = str(psutil.cpu_percent())
    speak('cpu is at' +usage)
    battery = psutil.sensors_battery()
    speak("battery is at")
    speak(battery.percent)
def jokes():
    speak(pyjokes.get_joke())
def Translate():
    speak("what I should Translate??")
    sentence = takeCommand().lower()
    trans = Translator()
    trans_sen = trans.translate(sentence,src='en',dest='mr')
    print(trans_sen.text)
    speak(trans_sen.text)
def news():
    newsapi = NewsApiClient(api_key='57aead5647ab48e0886c10dabe6196a7')
    data = newsapi.get_top_headlines(q='india',country='in',
                                      language='en',
                                      page_size=5)
    at = data['articles']
    for x,y in enumerate(at):
        print(f'{x} {y["description"]}')
        speak(f'{x} {y["description"]}')
    speak("that's it for now i'll updating you in some time ")
def get_wallpaper(path):
    access_key = 'A06xFEuSQ2KI-7K86B31yKXGWKRd_uQtHXEWs9adfFs'
    url = 'https://api.unsplash.com/photos/random?client_id=' + access_key
    params = {
        'query': 'HD wallpapers',
        'orientation': 'landscape'
    }
    response = requests.get(url, params=params).json()
    image_source = response['urls']['full']
    image = wget.download(image_source, path)
    return image
def change_wallpaper(path):
    wallpaper = get_wallpaper(path)
    ctypes.windll.user32.SystemParametersInfoW(SPI_SETDESKWALLPAPER, 0, path , 0)

class thread(threading.Thread):
    def __init__(self, thread_hr, thread_min):
        threading.Thread.__init__(self)
        self.thread_hr = thread_hr
        self.thread_min = thread_min
    def run(self):
        while True:
            if self.thread_hr==datetime.datetime.now().hour:
                if self.thread_min==datetime.datetime.now().minute:
                    print("Alarm in running")
                    winsound.PlaySound("abc", winsound.SND_LOOP)
                elif self.thread_min < datetime.datetime.now().minute:
                    break

class MainTread(QThread):
    def __init__(self):
        super(MainTread,self).__init__()
    def run(self):
        self.TaskExecution()
    def takeCommand(self):
        r = sr.Recognizer()
        with sr.Microphone() as source:
            print("Listening.....")
            r.pause_threshold = 1
            audio = r.listen(source)
        try:
            print("Recognizing....")
            query =r.recognize_google(audio, language ='en-in')
            print(query)
        except Exception as e:
            print(e)
            speak("")
            return "None"
        return query
    def TaskExecution(self):
        wishme()
        print(QThread)
        while True:
            self.query=self.takeCommand().lower()
            if 'roma' in self.query:
                self.query = self.query.replace('roma','')
                if 'time' in self.query:
                    time()
                elif 'date' in self.query:
                    date()
                elif 'about yourself' in self.query:
                    speak('"Roma is an artificial intelligence-
powered virtual assistant developed by Rupesh Ronak and Siddesh . "
"Programming language use to develope Kalki is python.""The assistant uses voice q
ueries and a naturallanguage user interface to answer questions, make recommendati
ons, and perform actions by delegating requests to a set of Internet services. "')
                elif 'wikipedia' in self.query: #wikipedia sachin tendulkar
                    try:
                        speak('searching....')
                        self.query=self.query.replace("wikipedia","")
                        result =wikipedia.summary(self.query , sentences=2)
                        print(result)
                        speak(result)
                    except Exception as e:
                        speak("Sorry Cannot able to find")
                elif 'search website in chrome' in self.query:
                    speak('what you want to search')
                    chromepath = "C:/Program Files/Google/Chrome/Application/chrom
e.exe %s"
                    search = takeCommand().lower()
                    wb.get(chromepath).open_new_tab(search+'.com')
                elif 'open youtube' in self.query:
                    wb.open("youtube.com")
                elif 'play' in self.query:
                    self.query=self.query.replace("play","")
                    speak('playing ' + self.query)
                    pywhatkit.playonyt(self.query)
                elif 'remember that' in self.query:
                    speak("what should i remember?")
                    data = takeCommand()
                    speak("you said me to remember that" +data)
                    path="C:/Jarvis/Data/"
                    filename = os.listdir("C:\\")
                    if 'Jarvis' in filename:
                        files=  os.listdir("C:\\Jarvis")
                        if "Data" in files:
                            remember =open('C:/Jarvis/Data/data.txt','w')
                            remember.write(data)
                        else:
                            os.mkdir("C:\\Jarvis\\Data")
                            remember =open('C:/Jarvis/Data/data.txt','w')
                            remember.write(data)
                    else :
                        os.makedirs("C:\\Jarvis\\Data")
                        remember =open('C:/Jarvis/Data/data.txt','w')
                        remember.write(data)
                    remember.close()
                elif 'forget anything' in self.query:
                    remember =open('C:/Jarvis/Data/data.txt', 'r')
                    speak("you said me remember that" +remember.read())
                elif 'screenshot' in self.query:
                    screenshot()
                    speak("done!")
                elif 'cpu' in self.query:
                    cpu()
                elif 'joke' in self.query:
                    jokes()
                elif 'whatsapp' in self.query:
                    wapath = "C:\\Users\\rupes\\AppData\\Local\\WhatsApp\\WhatsApp
.exe"
                    os.startfile(wapath)
                elif 'telegram' in self.query:
                    tgpath = "C:\\Users\\rupes\\AppData\\Roaming\\Telegram Desktop
\\Telegram.exe"
                    os.startfile(tgpath)
                elif 'google' in self.query: #google square root of 4
                    print('searching....')
                    speak('searching....')
                    try:
                        try:
                            self.query=self.query.replace("google","")
                            res=client.self.query(self.query)
                            results=next(res.results).text
                            print(results)
                            speak(results)
                        except:
                            results = wikipedia.summary(self.query, sentences=2)
                            print(results)
                            speak(results)
                    except:
                        wb.open('www.google.com')
                elif 'translate' in self.query: #translate
                    Translate()
                elif 'news' in self.query:
                    news()
                elif 'change wallpaper' in self.query:
                    path="C:/Jarvis/Wallpaper/wallpaper_"+datetime.datetime.now().
strftime("%m%d%Y_%H%M%S")+".jpg"
                    filename = os.listdir("C:\\")
                    if 'Jarvis' in filename:
                        files=  os.listdir("C:\\Jarvis")
                        if "Wallpaper" in files:
                            change_wallpaper(path)
                        else:
                            os.mkdir("C:\\Jarvis\\Wallpaper")
                            change_wallpaper(path)
                    else :
                        os.makedirs("C:\\Jarvis\\Wallpaper")
                        change_wallpaper(path)
                    speak('Successfully change.')
                elif 'thank you' in self.query:
                    speak("'Welcome sir.''Its my pleasure to serve you.'")
                elif 'break' in self.query:
                    speak("going offline for while")
                    sleep(60)
                elif 'offline' in self.query:
                    quit()
                elif 'close' in self.query:
                    os.system("taskkill /F /IM WhatsApp.exe")
                elif "alarm" in self.query:#set alarm #
                    speak("Tell Time")
                    try:
                        tt = takeCommand()
                        tt =tt.replace("set alarm to","")
                        tt=tt.replace(".","")
                        tt=tt.upper()
                        altime= str(datetime.datetime.now().strptime(tt,"%I:%M %p"
))

                        altime =altime[11:-3]
                        print(altime)
                        Horeal =altime[:2]
                        Horeal = int(Horeal)
                        Mireal = altime[3:5]
                        Mireal=int(Mireal)
                        print(f"Done, {tt}")
                        speak(f"Done, {tt}")
                        thread1 = thread(Horeal, Mireal)
                        thread1.start()
                    except Exception as e:
                        speak("Not able to set alarm please tell time in right for
mat For example 4 02 PM ")
                elif "how to" in self.query:#how to
                    speak("Search mode activates")
                    while True:
                        speak("What you want to know?")
                        how =takeCommand()
                        try:
                            if "close" in how:
                                speak("Mode deactivate!")
                                break
                            else:
                                max_results =1
                                how_to =search_wikihow(how,max_results)
                                assert len(how_to)==1
                                how_to[0].print()
                                speak(how_to[0].summary)
                        except Exception as e:
                            speak("Sorry,Unable to find.")
                elif "volume up" in self.query:
                    pyautogui.press("volumeup")
                elif "volume down" in self.query:
                    pyautogui.press("volumedown")
                elif "mute" in self.query:
                    pyautogui.press("volumemute")
                elif "unmute" in self.query:
                    pyautogui.press("volumemute")
                elif "location" in self.query:
                    speak("Let me check")
                    try:
                        ipAdd= requests.get('https://api.ipify.org').text
                        print(ipAdd)
                        url = "https://get.geojs.io/v1/ip/geo/" +ipAdd+'.json'
                        geo_request= requests.get(url)
                        geo_data =geo_request.json()
                        city= geo_data['city']
                        country = geo_data['country']
                        print(city,country)
                        speak(f"location{city}in{country}")
                    except Exception as e:
                        speak("sorry")
                        pass
                elif "camera" in self.query:
                    subprocess.run('start microsoft.windows.camera:', shell=True)
                elif "tab" in self.query:
                    pyautogui.keyDown("alt")
                    pyautogui.press("tab")
                    sleep(1)
                    pyautogui.keyUp("alt")
                elif 'logout' in self.query:
                    os.system("shutdown -l")
                elif 'restart' in self.query:
                    os.system("shutdown /r /t 1")
                elif 'shutdown' in self.query:
                    os.system("shutdown /s /t 1")
                elif 'check user' in self.query:
                    count, name = livefacerecognizer.checkUser()
                    if count != 0:
                        speak("Hello ")
                        speak(name)
                    if count == 0:
                        speak("Unable to identify user")
                elif 'who am i' in self.query:
                    livefacerecognizer.live()

startExecution=MainTread()

class Main(QMainWindow):
    def __init__(self):
        super().__init__()
        self.ui=Ui_MainWindow()
        self.ui.setupUi(self)
        self.ui.pushButton.clicked.connect(self.startTask)
        self.ui.pushButton_2.clicked.connect(self.close)
    def startTask(self):
        self.ui.movie=QtGui.QMovie("C:/Users/bipin/Documents/MCA/Mini_Project/Roma
/new.gif")
        self.ui.label.setMovie(self.ui.movie)
        self.ui.movie.start()
        timer=QTimer(self)
        timer.timeout.connect(self.showTime)
        timer.start(1000)
        startExecution.start()
    def showTime(self):
        current_time=QTime.currentTime()
        current_date=QDate.currentDate()
        label_time=current_time.toString('hh:mm:ss')
        label_date=current_date.toString(Qt.ISODate)
        self.ui.textBrowser.setText(label_date)
        self.ui.textBrowser_2.setText(label_time)

app=QApplication(sys.argv)
jarvis=Main()
jarvis.show()
exit(app.exec_())
```

RomaGUI.py

```
from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(1109, 909)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(10, 10, 1091, 891))
        self.label.setText("")
        self.label.setPixmap(QtGui.QPixmap("C:/Users/bipin/Documents/MCA/Mini_Proj
ect/Roma/new.gif"))
        self.label.setScaledContents(True)
        self.label.setObjectName("label")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(870, 660, 141, 51))
        self.pushButton.setStyleSheet("font: 75 10pt \"MS Shell Dlg 2\";\n"
"background-color: rgb(21, 101, 132);")
        self.pushButton.setObjectName("pushButton")
        self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_2.setGeometry(QtCore.QRect(870, 730, 141, 51))
        self.pushButton_2.setStyleSheet("font: 75 10pt \"MS Shell Dlg 2\";\n"
"background-color: rgb(21, 101, 132);")
        self.pushButton_2.setObjectName("pushButton_2")
        self.textBrowser = QtWidgets.QTextBrowser(self.centralwidget)
        self.textBrowser.setGeometry(QtCore.QRect(800, 30, 256, 41))
        self.textBrowser.setAutoFillBackground(False)
        self.textBrowser.setStyleSheet("background:transparent;\n"
"border-color: rgb(33, 150, 200);\n"
"color:white;\n"
"font-size:20px ;" )
        self.textBrowser.setObjectName("textBrowser")
        self.textBrowser_2 = QtWidgets.QTextBrowser(self.centralwidget)
        self.textBrowser_2.setGeometry(QtCore.QRect(800, 100, 256, 41))
        self.textBrowser_2.setStyleSheet("background:transparent;\n"
"border-color: rgb(33, 150, 200);\n"
"color:white;\n"
"font-size:20px ;")
        self.textBrowser_2.setObjectName("textBrowser_2")
        MainWindow.setCentralWidget(self.centralwidget)
        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
        self.pushButton.setText(_translate("MainWindow", "RUN"))
        self.pushButton_2.setText(_translate("MainWindow", "EXIT"))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

FaceRecognition.py

```
import cv2
import os
import numpy as np

name = {0: "Unknown",1: "Ronak",2: "Rupesh",3: "Siddhesh"}
#Creating Training Modle
def createTrainingModle():
    faces, faceID = labels_for_training("faceRecognition_master/TrainingImages")
    face_recognizer = train_classifier(faces, faceID)
    face_recognizer.write("faceRecognition_master/TrainingModle/trainingData.yml")

    return face_recognizer
#Use Created Training Model
def useTrainingModle():
    face_recognizer = cv2.face.LBPHFaceRecognizer_create()
    face_recognizer.read("faceRecognition_master/TrainingModle/trainingData.yml")
    return face_recognizer
def faceDetection(test_img):
    gray_img = cv2.cvtColor(test_img,cv2.COLOR_BGR2GRAY) #convert color image to g
rayscle img
    face_haar_cascade = cv2.CascadeClassifier("faceRecognition_master/HaarCascade/
haarcascade_frontalface_default.xml")
    faces = face_haar_cascade.detectMultiScale(gray_img,scaleFactor=1.32,minNeighb
ors=5)
    return faces,gray_img
def labels_for_training(directory):
    faces = []
    faceID = []
    for path,subdirname,filenames in os.walk(directory):
        for filename in filenames:
            if filename.startswith("."):
                print("Skippping system file")
                continue
            id = os.path.basename(path)
            img_path = os.path.join(path,filename)
            print('img_path: ',img_path)
            #print("id",id)
            test_img = cv2.imread(img_path)
            if test_img is None:
                print("Image not loaded properly")
                continue
            faces_rect,gray_img = faceDetection(test_img)
            if len(faces_rect)!=1:
                continue
            (x,y,w,h) = faces_rect[0]
            roi_gray = gray_img[y:y+w,x:x+h]
            faces.append(roi_gray)
            faceID.append(int(id))
    return faces,faceID
def train_classifier(faces,faceID):
    face_recognizer = cv2.face.LBPHFaceRecognizer_create()
    face_recognizer.train(faces,np.array(faceID))
    return face_recognizer
def draw_rect(test_img,face):
    (x,y,w,h) = face
    cv2.rectangle(test_img,(x,y),(x+w,y+h),(255,0,0),thickness=4)
def put_text(test_img,text,x,y):
    cv2.putText(test_img,text,(x,y),cv2.FONT_HERSHEY_DUPLEX,2,(255,0,0),4)
```

Trainer.py

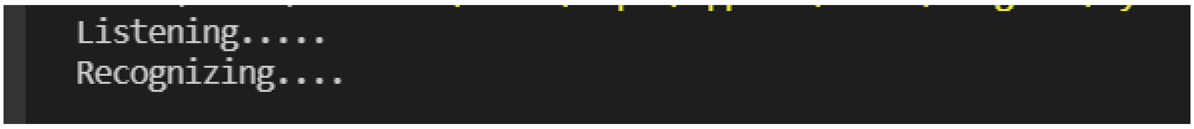
```
import cv2
import os
import numpy as np
import FaceRecognition as fr

#This module take images stored in disk and perform face detection
test_img= cv2.imread("faceRecognition_master/TestImages/Ronak.jpg")
faces_detected,gray_img = fr.faceDetection(test_img)
print("faceDetected: ",faces_detected)
filename = os.listdir("faceRecognition_master/TrainingModle")
if 'trainingData.yml' in filename:
    print("Skipping Training Modle Creation")
    face_recognizer = fr.useTrainingModle()
else:
    print("Creating Training Modle....")
    face_recognizer = fr.createTrainingModle()
for face in faces_detected:
    (x,y,w,h) = face
    roi_gray = gray_img[y:y+h,x:x+h]
    label,confidence = face_recognizer.predict(roi_gray)
    confidence =np.absolute(confidence-100)
    print("confidence: ",confidence)
    print("label ",fr.name[label] )
    fr.draw_rect(test_img,face)
    predicted_name = fr.name[label]
    if(confidence<40):
        continue
    fr.put_text(test_img,predicted_name,x,y)
resized_img = cv2.resize(test_img,(1000,1000))
cv2.imshow("faces detection tutorial",resized_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

LiveFaceRecognizer.py

```
import cv2
import os
import numpy as np
from faceRecognition_master import FaceRecognition as fr

face_recognizer = fr.useTrainingModle()  #Load Traning Modle

def live():
    if face_recognizer is np.empty:
        return "No Traning Modle Found. "
        exit(0)
    cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)   #capture image from web cam
    while True:
        ret, test_img = cap.read()
        faces_detected,gray_img = fr.faceDetection(test_img)
        for face in faces_detected:
            (x,y,w,h) = face
            roi_gray = gray_img[y:y+w,x:x+h]
            label,confidence = face_recognizer.predict(roi_gray)
            confidence =np.absolute(confidence-100)
            #print("Conficdence ", confidence)
            #print("Lable", lable)
            fr.draw_rect(test_img,face)
            predicted_name = fr.name[label]
            fr.put_text(test_img,predicted_name,x,y)
            if confidence > 60:
                fr.put_text(test_img,predicted_name,x,y)
        resized_img = cv2.resize(test_img,(1000,700))
        cv2.imshow("FaceRecognition",resized_img)
        if cv2.waitKey(10) == ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()

lable =0
def checkUser():

    if face_recognizer is np.empty:
        return "No Traning Modle Found. "
        exit(0)
    cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    i = 0
    a = 0
    while a<30:
        a=a+1
        ret,test_img = cap.read()
        faces_detected,gray_img = fr.faceDetection(test_img)
        cv2.waitKey(10)
        resized_img = cv2.resize(test_img,(1000,700))
        for face in faces_detected:
            (x,y,w,h) = face
            roi_gray = gray_img[y:y+h,x:x+w]
            label,confidence = face_recognizer.predict(roi_gray)
            confidence =np.absolute(confidence-100)
            #print("confidence: ",confidence)
            #print("label: ",fr.name[label])
            predicted_name = fr.name[label]
            a=a+1
            if confidence > 50:
                i=i+1
        if cv2.waitKey(10) == ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows
    if i >= 10:
        return 1, fr.name[label]
    else:
        a = "Unknown "
        return 0, a
```

RESULT

Start the Assistant & Accept request from user :-

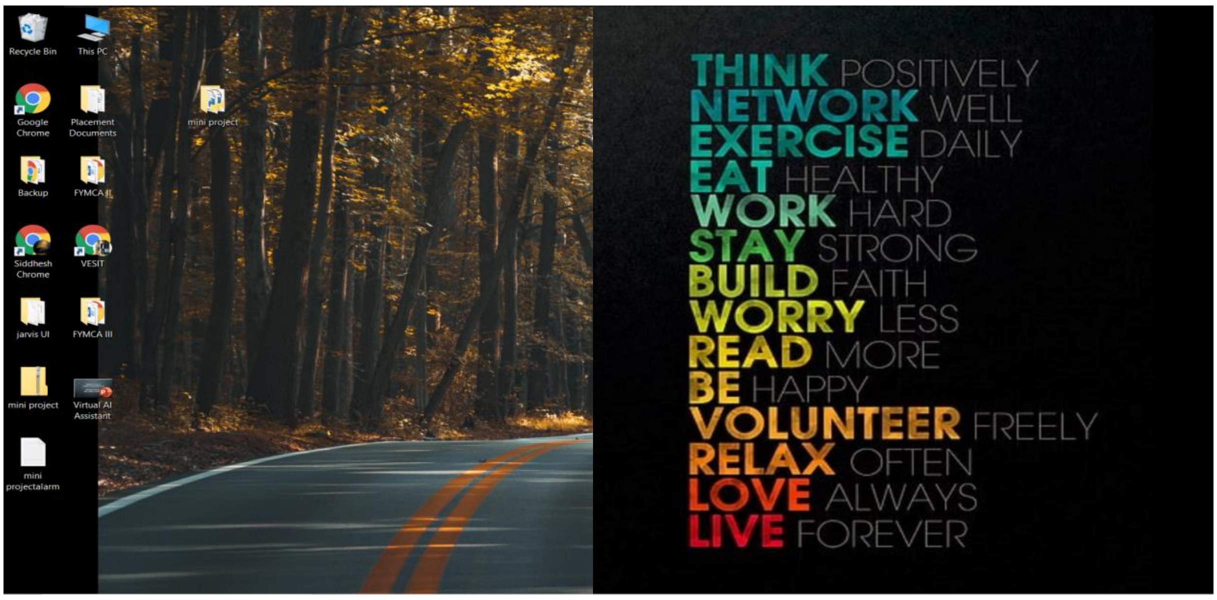


```
Listening.....
Recognizing....
```

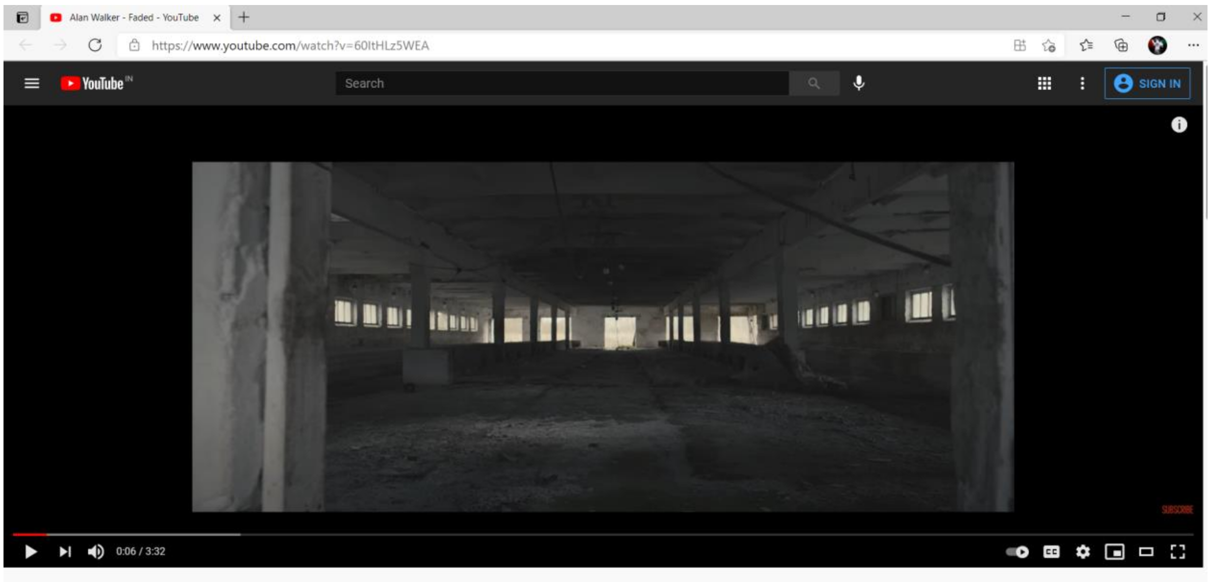
Change Wallpaper :-

Old Wallpaper

New Wallpaper

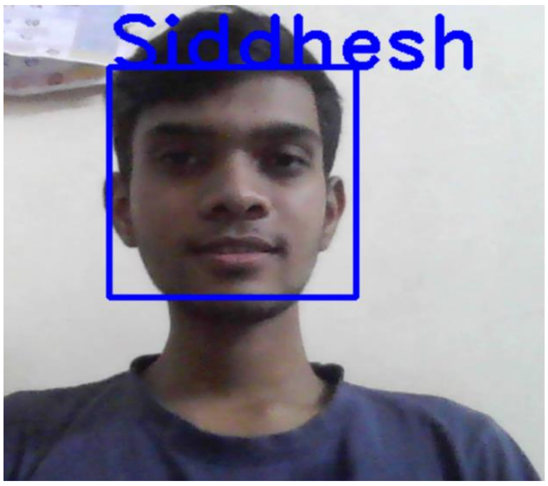


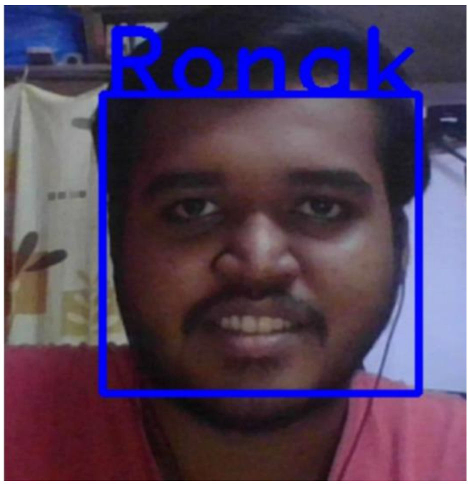
Play Songs :-



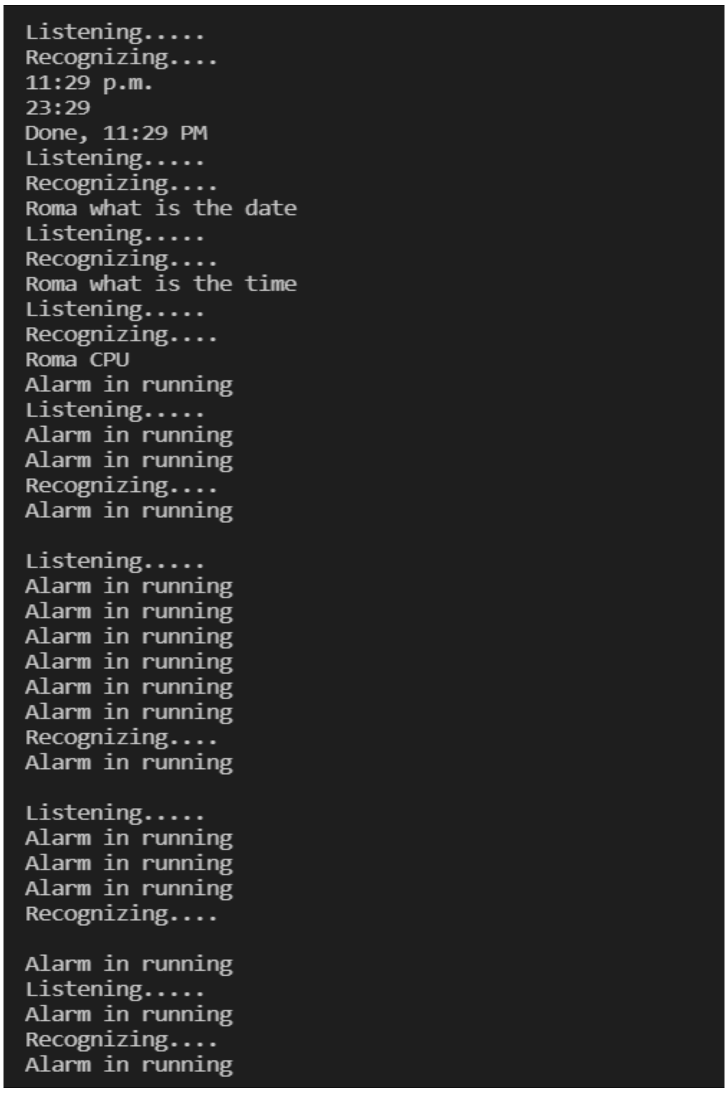
Alan Walker - Faded - YouTube
https://www.youtube.com/watch?v=60ItHLz5WEA
YouTube
Search
SIGN IN
0:06 / 3:32

Face recognition



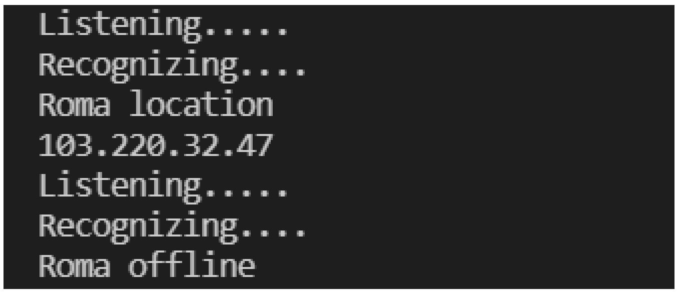


Alarm

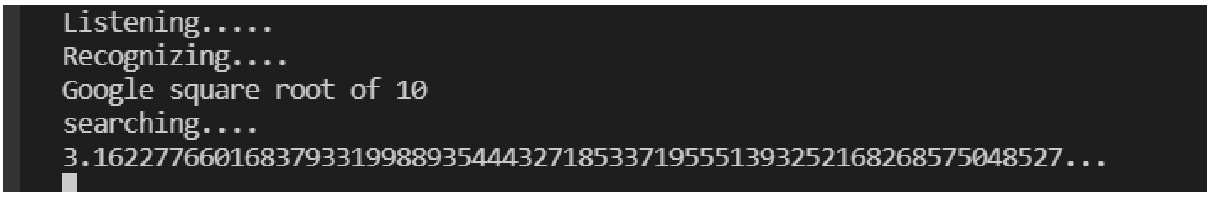


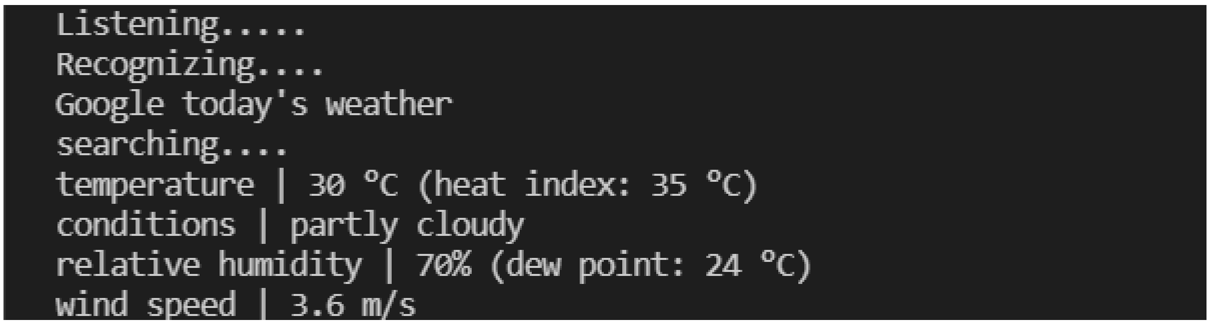
```
Listening.....
Recognizing....
11:29 p.m.
23:29
Done, 11:29 PM
Listening.....
Recognizing....
Roma what is the date
Listening.....
Recognizing....
Roma what is the time
Listening.....
Recognizing....
Roma CPU
Alarm in running
Listening.....
Alarm in running
Alarm in running
Recognizing....
Alarm in running

Listening.....
Alarm in running
Alarm in running
Alarm in running
Alarm in running
Alarm in running
Alarm in running
Recognizing....
Alarm in running

Listening.....
Alarm in running
Alarm in running
Alarm in running
Recognizing....

Alarm in running
Listening.....
Alarm in running
Recognizing....
Alarm in running
```

Current Location

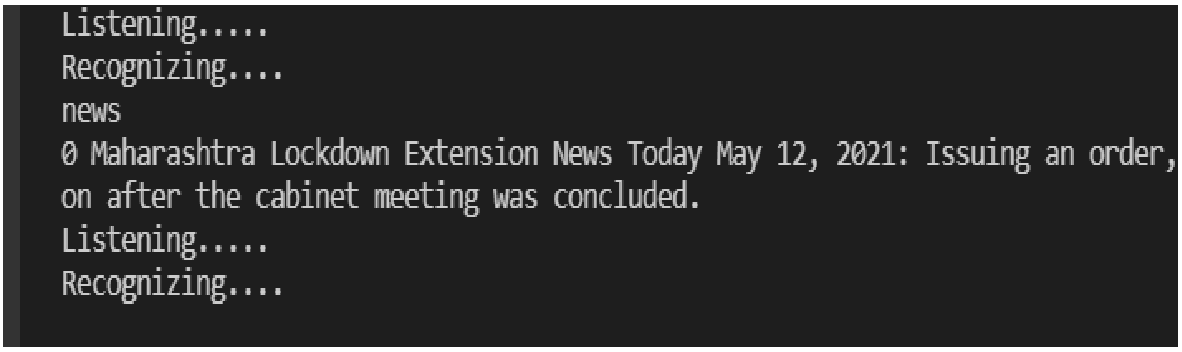

```
Listening.....
Recognizing....
Roma location
103.220.32.47
Listening.....
Recognizing....
Roma offline
```

Answering the user questions

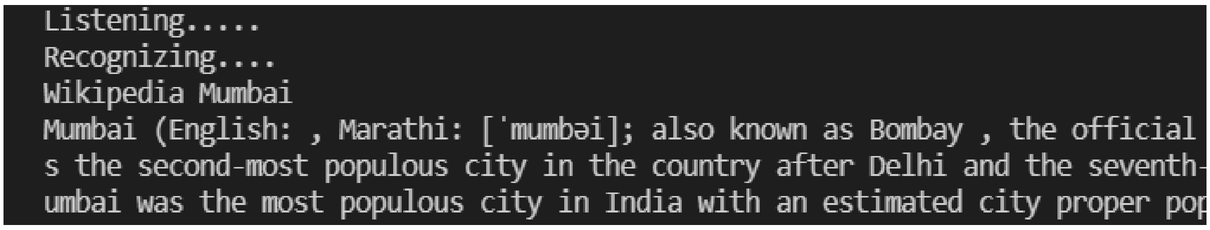

```
Listening.....
Recognizing....
Google square root of 10
searching....
3.1622776601683793319988935444327185337195551393252168268575048527...
```


```
Listening.....
Recognizing....
Google today's weather
searching....
temperature | 30 °C (heat index: 35 °C)
conditions | partly cloudy
relative humidity | 70% (dew point: 24 °C)
wind speed | 3.6 m/s
```

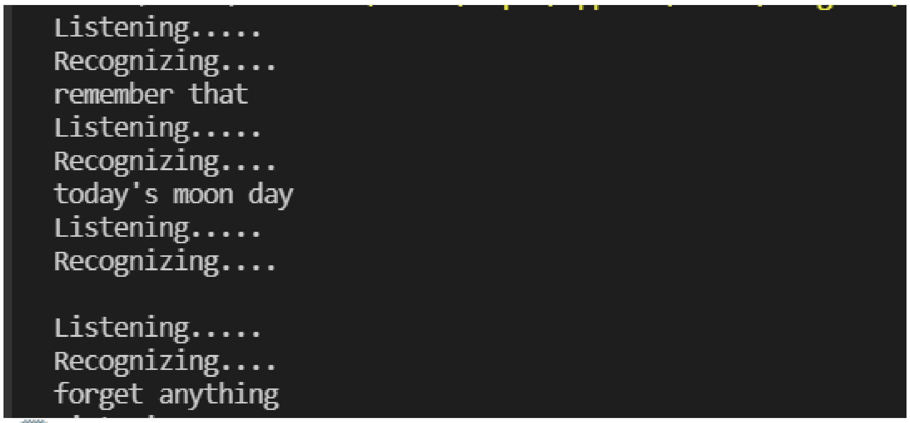
News


```
Listening.....
Recognizing....
news
0 Maharashtra Lockdown Extension News Today May 12, 2021: Issuing an order,
on after the cabinet meeting was concluded.
Listening.....
Recognizing....
```

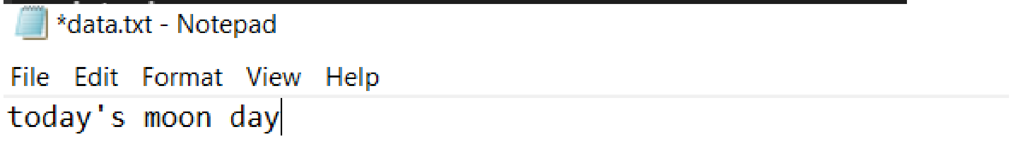
Wikipedia


```
Listening.....
Recognizing....
Wikipedia Mumbai
Mumbai (English: , Marathi: [ˈmumbəi]; also known as Bombay , the official
s the second-most populous city in the country after Delhi and the seventh-
umbai was the most populous city in India with an estimated city proper pop
```

Remember Function



```
Listening.....
Recognizing....
remember that
Listening.....
Recognizing....
today's moon day
Listening.....
Recognizing....

Listening.....
Recognizing....
forget anything
```

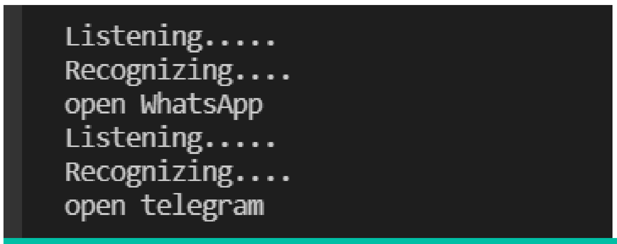


*data.txt - Notepad

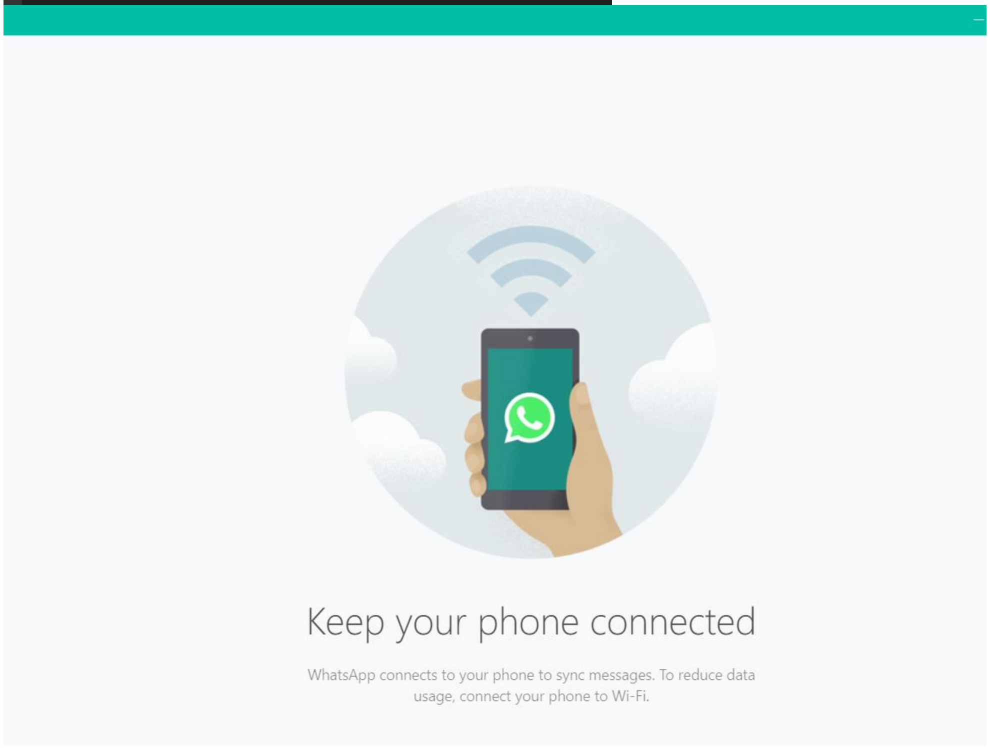
File Edit Format View Help

```
today's moon day
```

Open Apps from PC



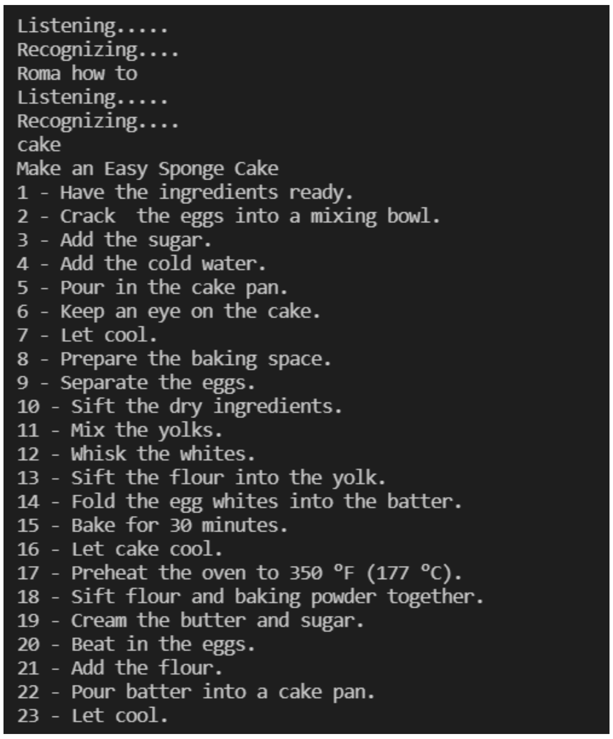
```
Listening.....
Recognizing....
open WhatsApp
Listening.....
Recognizing....
open telegram
```



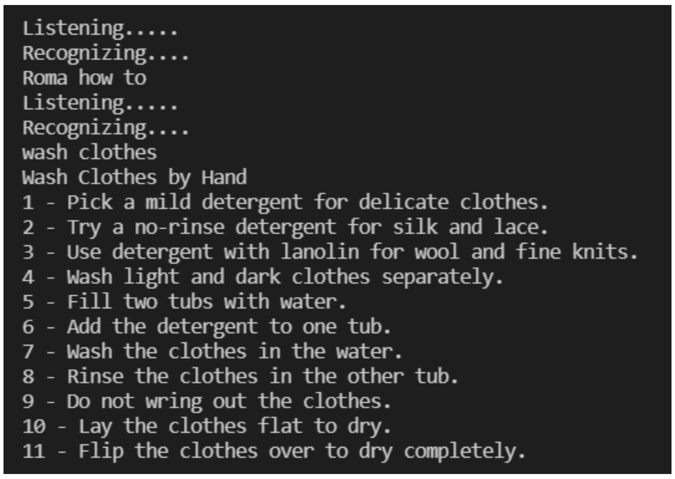
Keep your phone connected

WhatsApp connects to your phone to sync messages. To reduce data usage, connect your phone to Wi-Fi.

Get steps for execution of day to day tasks

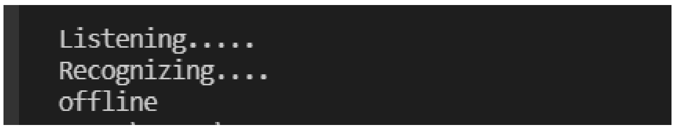


```
Listening.....
Recognizing....
Roma how to
Listening.....
Recognizing....
cake
Make an Easy Sponge Cake
1 - Have the ingredients ready.
2 - Crack  the eggs into a mixing bowl.
3 - Add the sugar.
4 - Add the cold water.
5 - Pour in the cake pan.
6 - Keep an eye on the cake.
7 - Let cool.
8 - Prepare the baking space.
9 - Separate the eggs.
10 - Sift the dry ingredients.
11 - Mix the yolks.
12 - Whisk the whites.
13 - Sift the flour into the yolk.
14 - Fold the egg whites into the batter.
15 - Bake for 30 minutes.
16 - Let cake cool.
17 - Preheat the oven to 350 °F (177 °C).
18 - Sift flour and baking powder together.
19 - Cream the butter and sugar.
20 - Beat in the eggs.
21 - Add the flour.
22 - Pour batter into a cake pan.
23 - Let cool.
```



```
Listening.....
Recognizing....
Roma how to
Listening.....
Recognizing....
wash clothes
Wash Clothes by Hand
1 - Pick a mild detergent for delicate clothes.
2 - Try a no-rinse detergent for silk and lace.
3 - Use detergent with lanolin for wool and fine knits.
4 - Wash light and dark clothes separately.
5 - Fill two tubs with water.
6 - Add the detergent to one tub.
7 - Wash the clothes in the water.
8 - Rinse the clothes in the other tub.
9 - Do not wring out the clothes.
10 - Lay the clothes flat to dry.
11 - Flip the clothes over to dry completely.
```

Shutdown Assistant



```
Listening.....
Recognizing....
offline
```

Future Enhancement & Conclusion

- **Future Enhancement**
    1. Generating .exe file for easier distribution on different platforms
    2. Integrate with additional functionalities (HOME AUTOMATION)
    3. Integrate with IOT to function it as ALEXA

- **Conclusion**

  Through this voice assistant, we have automated various services using a single line command. It eases most of the tasks of the user like searching the web, retrieving latest news, further more. We aim to make this project a complete assistant and make it smart enough to act as the query given.

REFERENCE

- **Websites referred**
  - www.stackoverflow.com
  - www.pythonprogramming.net
  - www.codecademy.com
  - www.tutorialspoint.com
  - www.google.co.in

- **Books referred**
  - Python Programming - Kiran Gurbani
  - Learning Python - Mark Lutz

- **YouTube Channels referred**
  - CS Dojo
  - edureka!

- **Documents referred**
  - Designing Personal Assistant Software for Task Management using Semantic Web Technologies and Knowledge Databases
    - Purushotham Botla
  - Python code for Artificial Intelligence: Foundations of Computational Agents
    - David L. Poole and Alan K. Mackworth

THANK YOU