PROJECT

On

"J.A.R.V.I.S"

SUBMITTED TO

J.C. Bose University of Science and Technology, YMCA

<u>IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF</u>

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

ARAVALi
COLLEGE OF ENGINEERING & MANAGEMENT
creating achievers

Submitted To: **Submitted By:**

Department of Computer Science & Engineering
(2021-22)

DECLARATION

I/We hereby declare that the project work entitled **“J.A.R.V.I.S”** submitted to J.C.Bose University Of Science and Technology, Faridabad, Haryana (India), is a record of an original work done by me/us under the guidance of

“Ms. Anshu Sharma” (Assistant Professor) in Computer Science and Engineering, ARAVALI COLLEGE OF ENGINEERING AND MANAGEMENT, FARIDABAD, and this project work is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING.

CERTIFICATE

Certified that this project report **"J.A.R.V.I.S"** is the bonafide work of "**Group 35**" who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
HEAD OF THE DEPARTMENT	
Name:	**DESIGNATION(CSE)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
ARAVALI COLLEGE OF ENGINEERING AND MANAGEMENT
FARIDABAD – 121002

ACKNOWLEDGEMENT

This project would not have taken shape, without the guidance provided by **"Ms. Anshu Sharma"**, our Trainer who helped in the modules of our project and resolved all the technical as well as other problems related to the project and, for always providing us with a helping hand whenever we faced any bottlenecks, inspite of being quite busy with their hectic schedules.

We would also like to thank our project supervisor **……………………………….** who gave us the opportunity and provided us all the academic and conceptual support for our project.

Above all we wish to express our heartfelt gratitude to **…………………………..**, H.O.D, CSE DEPARTMENT whose support has greatly boosted our self-confidence and will go a long way on helping us to reach further milestones and greater heights.

ABSTRACT

The main objective of this project is to design a voice based ai desktop assistant.

Then, to workout all the necessary functions with help of voice based commands.

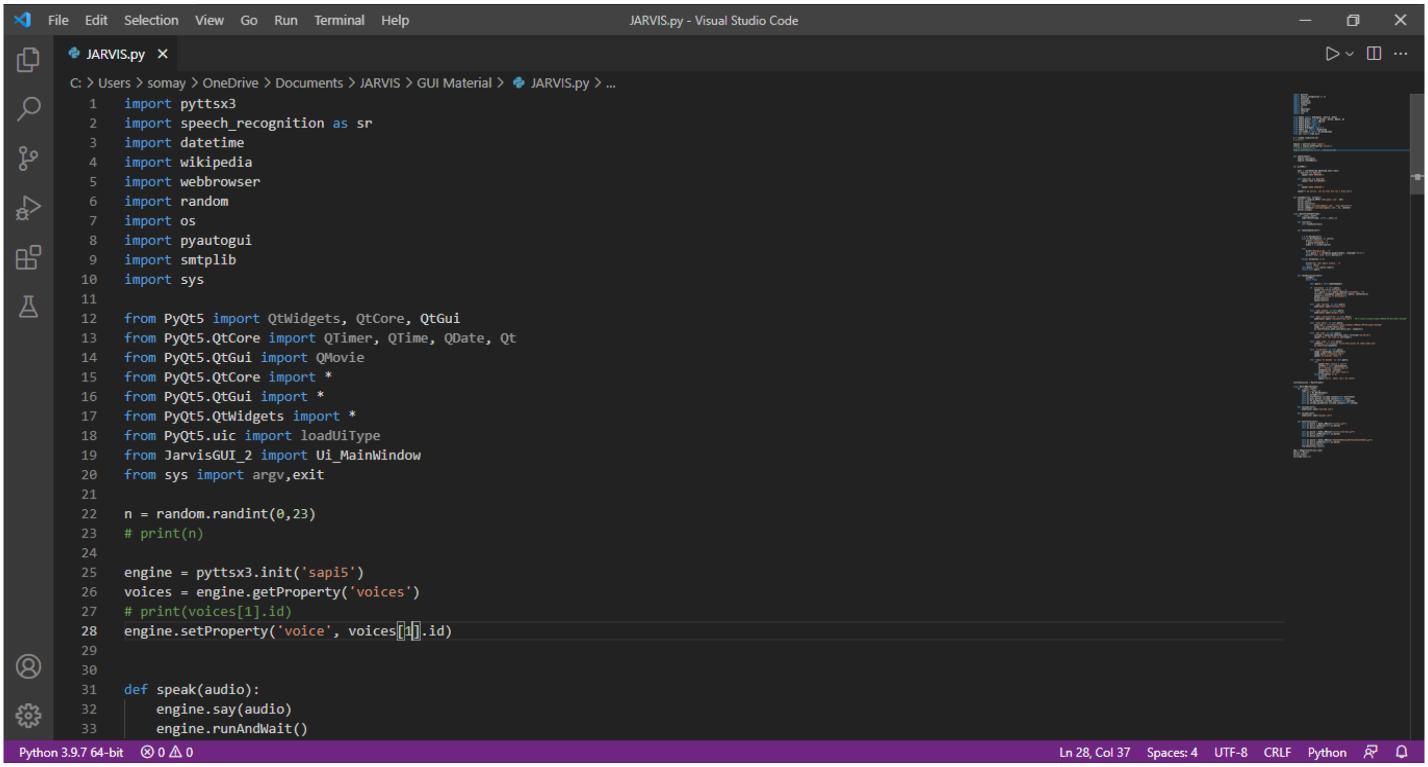
This project includes an implementation of an intelligent voice recognition assistant for laptop OS where functionality on current existing applications on other platforms is compared. Until this day, there has not been any good alternative for laptop OS, so this project aims to implement a voice assistant for the laptop OS platform while describing the difficulties and challenges that lies in this task.

Functionalities of this project include:

1. It Can Send Emails.
2. It Can Read Pdf.
3. It Can Open Command Prompt, Your Favorite Ide, Notepad Etc.
4. It Can Play Music.
5. It Can Do Wikipedia Searches For You.
6. It Can Open Websites Like Google, Youtube, Etc., In A Web Browser.
7. It Can Give Weather Forecast.
8. It Can Give Desktop Reminders Of Your Choice.

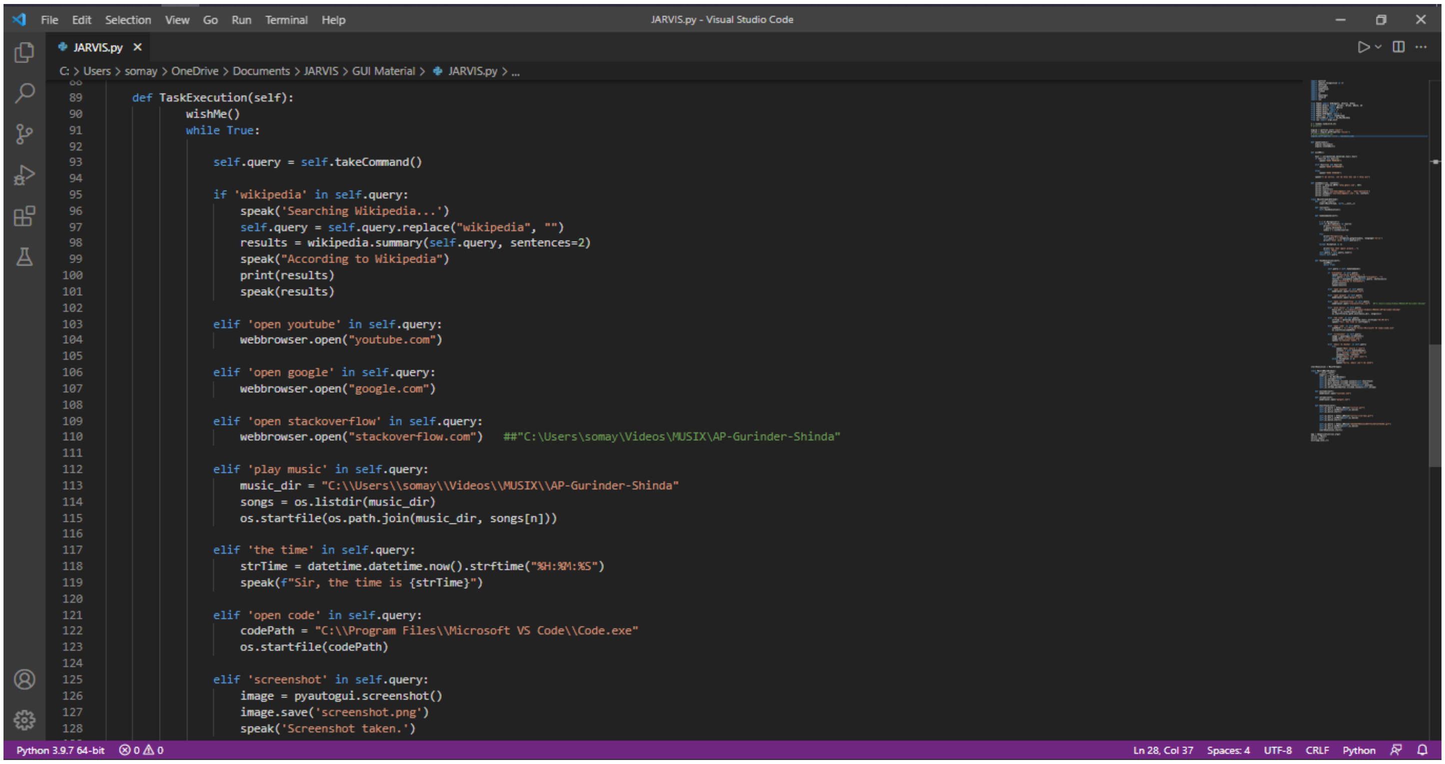
LIST OF FIGURES

It depicts the Number And Types of Libraries Involved:


File Edit Selection View Go Run Terminal Help
JARVIS.py - Visual Studio Code
JARVIS.py
C: > Users > somay > OneDrive > Documents > JARVIS > GUI Material > JARVIS.py > ...
Python 3.9.7 64-bit ⊗ 0 ⚠ 0
Ln 28, Col 37 Spaces: 4 UTF-8 CRLF Python

```python
import pyttsx3
import speech_recognition as sr
import datetime
import wikipedia
import webbrowser
import random
import os
import pyautogui
import smtplib
import sys

from PyQt5 import QtWidgets, QtCore, QtGui
from PyQt5.QtCore import QTimer, QTime, QDate, Qt
from PyQt5.QtGui import QMovie
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtWidgets import *
from PyQt5.uic import loadUiType
from JarvisGUI_2 import Ui_MainWindow
from sys import argv,exit

n = random.randint(0,23)
# print(n)

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


def speak(audio):
    engine.say(audio)
    engine.runAndWait()
```

It depicts the Task Execution Commands for proper Functioning:


File Edit Selection View Go Run Terminal Help
JARVIS.py - Visual Studio Code
JARVIS.py
C: > Users > somay > OneDrive > Documents > JARVIS > GUI Material > JARVIS.py > ...
Python 3.9.7 64-bit ⊗ 0 ⚠ 0
Ln 28, Col 37 Spaces: 4 UTF-8 CRLF Python

```python
    def TaskExecution(self):
            wishMe()
            while True:

                self.query = self.takeCommand()

                if 'wikipedia' in self.query:
                    speak('Searching Wikipedia...')
                    self.query = self.query.replace("wikipedia", "")
                    results = wikipedia.summary(self.query, sentences=2)
                    speak("According to Wikipedia")
                    print(results)
                    speak(results)

                elif 'open youtube' in self.query:
                    webbrowser.open("youtube.com")

                elif 'open google' in self.query:
                    webbrowser.open("google.com")

                elif 'open stackoverflow' in self.query:
                    webbrowser.open("stackoverflow.com")   ##"C:\Users\somay\Videos\MUSIX\AP-Gurinder-Shinda"

                elif 'play music' in self.query:
                    music_dir = "C:\\Users\\somay\\Videos\\MUSIX\\AP-Gurinder-Shinda"
                    songs = os.listdir(music_dir)
                    os.startfile(os.path.join(music_dir, songs[n]))

                elif 'the time' in self.query:
                    strTime = datetime.datetime.now().strftime("%H:%M:%S")
                    speak(f"Sir, the time is {strTime}")

                elif 'open code' in self.query:
                    codePath = "C:\\Program Files\\Microsoft VS Code\\Code.exe"
                    os.startfile(codePath)

                elif 'screenshot' in self.query:
                    image = pyautogui.screenshot()
                    image.save('screenshot.png')
                    speak('Screenshot taken.')
```

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

Artificial intelligence when used with machines, it shows us the capability of thinking like humans. In this, a computer system is designed in such a way that typically requires interaction from human. As we know python is an emerging language so it becomes easy to write a script for voice assistant in python. The instructions for the assistant can be handled as per the requirement of user. Speech recognition is the alexa, siri, etc. In python there is an api called speech recognition which allows us to convert speech into text. It was an interesting task to make my own assistant. It became easier to send emails without typing any word,

Searching on google without opening the browser, and performing many other daily tasks like playing music, opening your favorite ide with the help of a single voice command. In the current scenario, advancement in technologies are such that they can perform any task with same effectiveness or can say more effectively than us. By making this project, i realized that the concept of ai in every field is decreasing human effort and saving time. As the voice assistant is using artificial intelligence hence the result that it is providing are highly accurate and efficient. The assistant can help to reduce human effort and consumes time while performing any task, they removed the concept of typing completely and behave as another individual to whom we are talking and asking to perform task. The assistant is no less than a human assistant but we can say that this is more effective and efficient to perform any task.

The libraries and packages used to make this assistant focuses on the time complexities and reduces time. The functionalities include , it can send emails, it can read pdf, it can send text on whatsapp, it can open command prompt, your favorite ide, notepad etc., It can play music, it can do wikipedia searches for you, it can open websites like google, youtube, etc., In a web browser, it can give weather forecast, it can give desktop reminders of your choice. It can have some basic conversation.

2. SYSTEM ANALYSIS

Platform

- **Operating System:** Windows 10 And Debian OS
- **Technologies Used:**

Front End: PyQt5

Processing language: Python

Dataset: ➢ SQLite
➢ Cookies

Software Requirements

1. Visual Studio
2. Console(EUDC Fonts)
3. Python Ide

Hardware Requirements

- Minimum Processor I3 Required
- Hard Drive Minimum 500gb
- RAM Minimum 4GB
- Stable Internet Connection

3. SYSTEM DESIGN

BASIC MODULES

There Following Modules In Desktop Voice Assistant.

Pyttsx3

Pyttsx3 is a text-to-speech conversion library in python. Unlike alternative libraries, it works offline, and is compatible with both python 2 and 3.

SPEECH RECOGNITION

Speech Recognition Is An Important Feature In Several Applications Used Such As Home Automation, Artificial Intelligence, Etc. This Article Aims To Provide An Introduction On How To Make Use Of The Speech Recognition Library Of Python.

DATE TIME

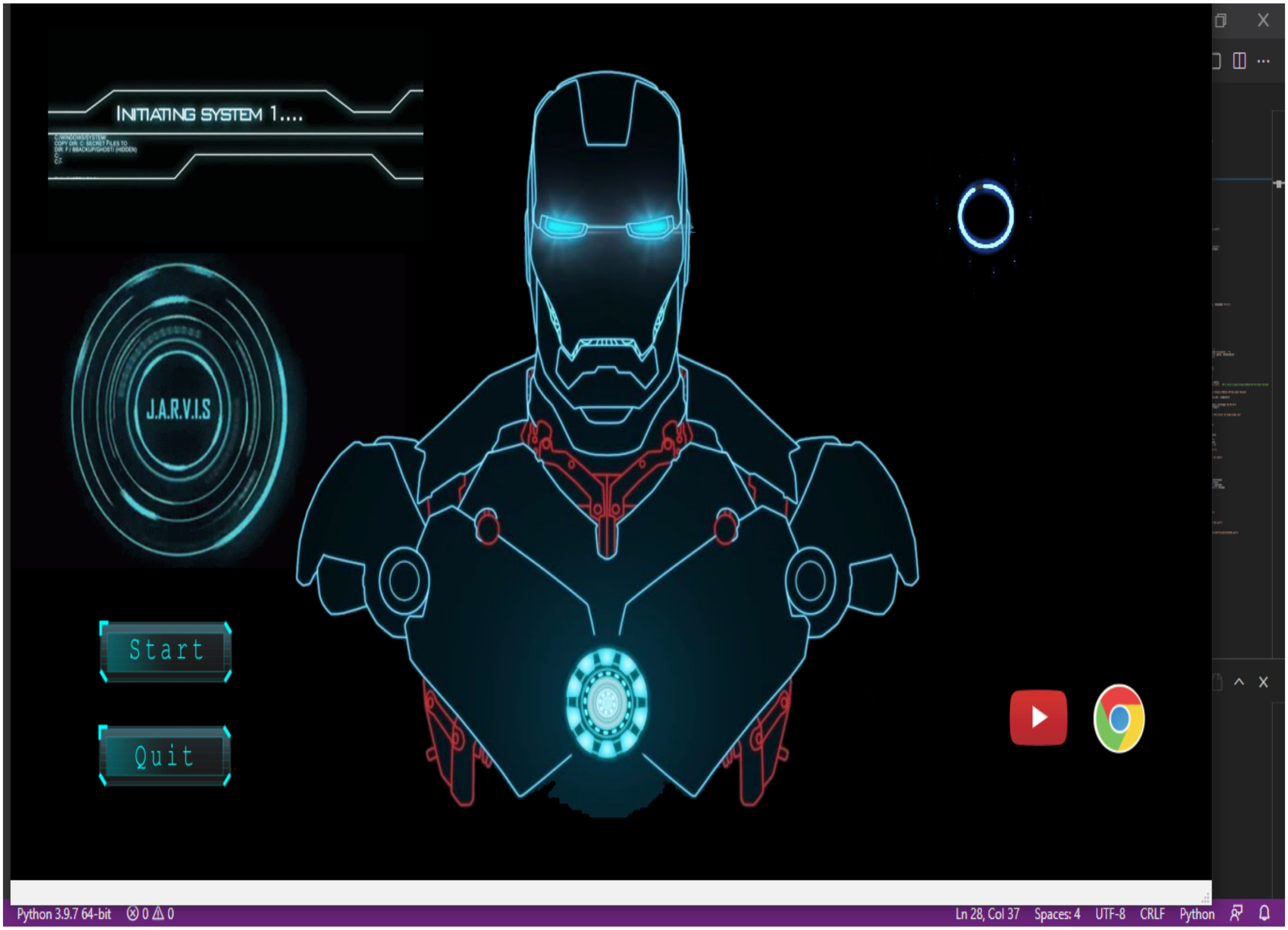
The date time module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. For related functionality, see also the time and calendar modules.

WIKIPEDIA

Wikipedia is A Python Library That Makes It Easy To Access And Parse Data From Wikipedia. Search Wikipedia, Get Article Summaries, Get Data Like Links And Images From A Page, And More. Wikipedia Wraps The media Wiki Api so You Can Focus On Using Wikipedia Data, Not Getting It.

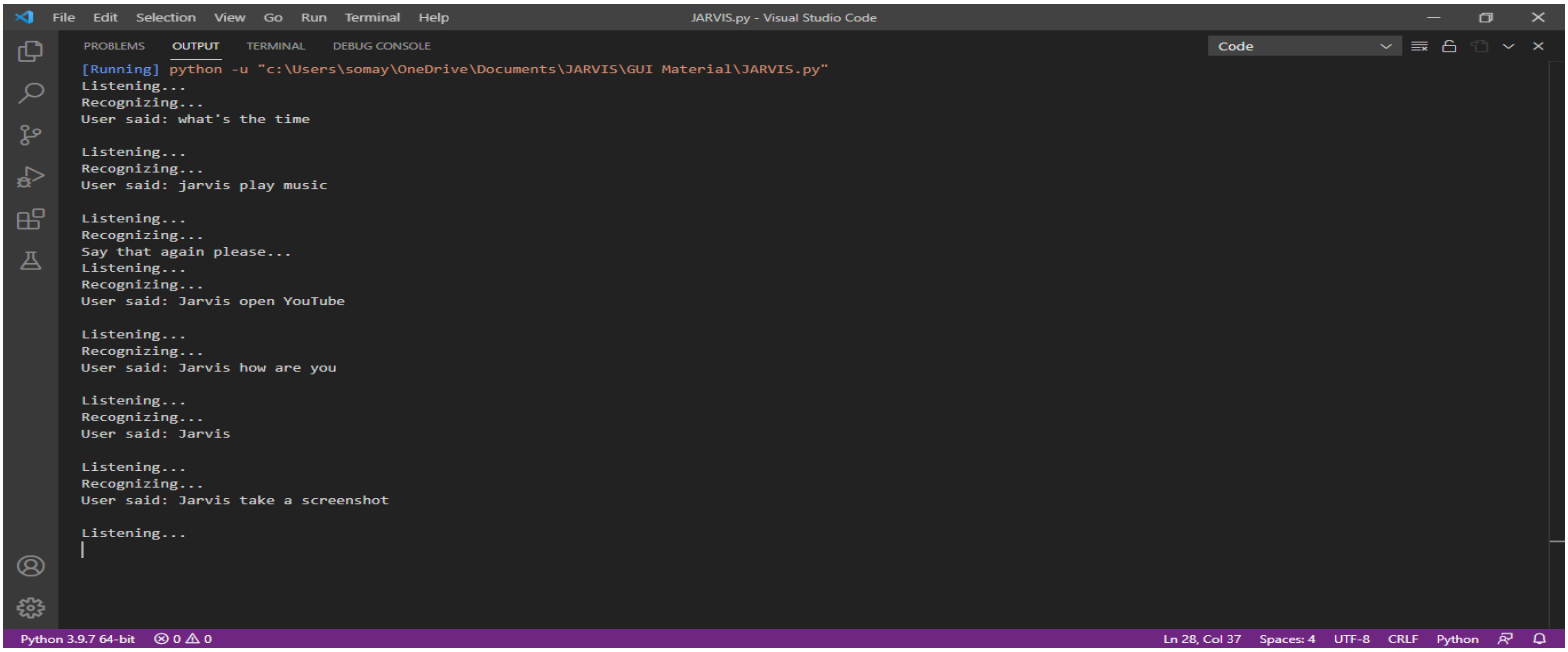
4. TESTING & TEST RESULTS

•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 THE QUERY MADE.



• ACCURACY TEST ID: T2 TEST PRIORITY: HIGH TEST OBJECTIVE:

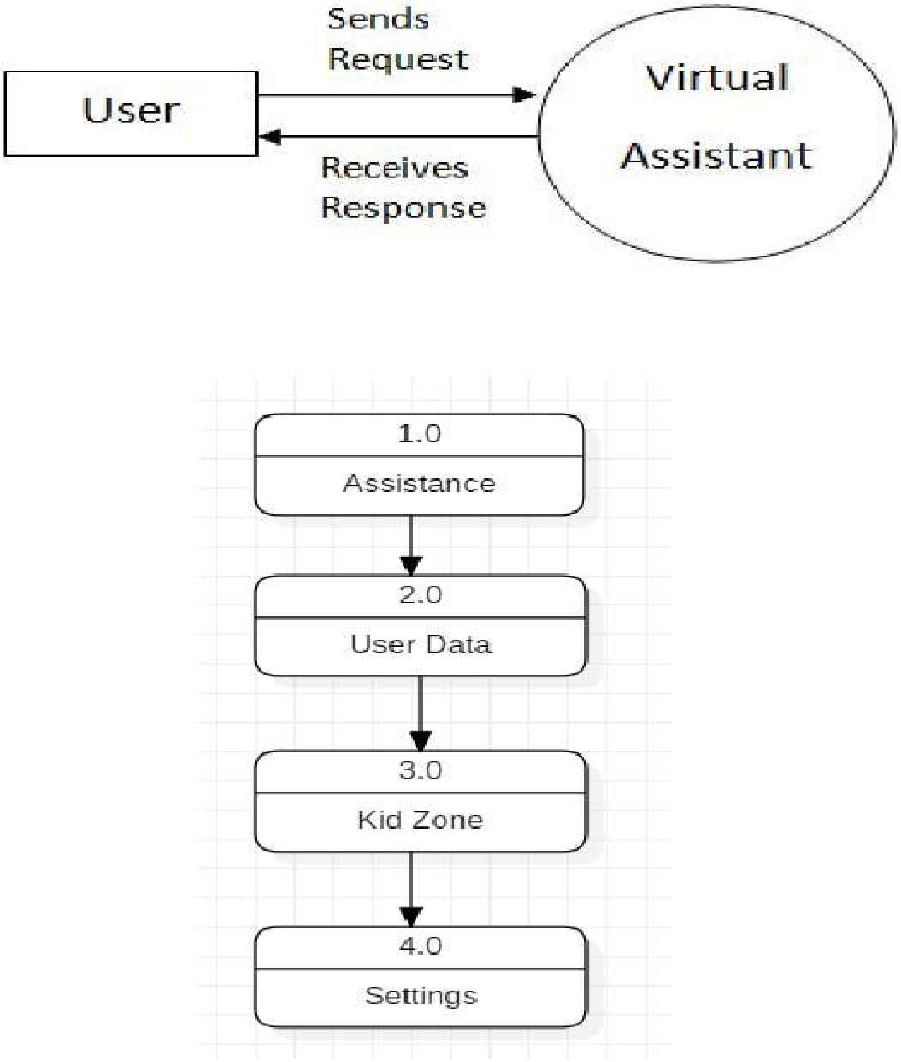
TO ASSURE THAT ANSWERS RETRIEVED BY SYSTEM ARE ACCURATE AS PER GATHERED DATA. DESCRIPTION: A VIRTUAL ASSISTANT SYSTEM IS MAINLY USED TO GET PRECISE ANSWERS TO ANY QUESTION ASKED. GETTING ANSWER IN A MOMENT IS OF NO USE IF THE ANSWER IS NOT CORRECT. ACCURACY IS OF UTMOST IMPORTANCE IN A VIRTUAL ASSISTANT SYSTEM.



```
[Running] python -u "c:\Users\somay\OneDrive\Documents\JARVIS\GUI Material\JARVIS.py"
Listening...
Recognizing...
User said: what's the time

Listening...
Recognizing...
User said: jarvis play music

Listening...
Recognizing...
Say that again please...
Listening...
Recognizing...
User said: Jarvis open YouTube

Listening...
Recognizing...
User said: Jarvis how are you

Listening...
Recognizing...
User said: Jarvis

Listening...
Recognizing...
User said: Jarvis take a screenshot

Listening...
```

• 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 AND NOT THE ACCURATE VALUE. GETTING EXACT VALUE IN SUCH CASES IS UNDESIRABLE.

5. SYSTEM IMPLEMENTATION



6. SUMMARY AND CONCLUSIONS

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 weather forecast details, vocabulary help and medical related queries. We aim to make this project a complete server assistant and make it smart enough to act as a replacement for a general server administration. The future plans include integrating jarvis with mobile using react native to provide a synchronised experience between the two connected devices. Further, in the long run, jarvis is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general server administrator does. The functionality would be seamless enough to replace the server administrator with jarvis.

FUTURE PROSPECTIVE

We plan to integrate JARVIS with mobile using react native, to provide a synchronized experience between the two connected devices. Further, in the long run, JARVIS is planned to feature auto deployment supporting elastic beanstalk, backup files, and all operations which a general server administrator does. The functionality would be seamless enough to replace the server administrator with JARVIS.

REFERENCES

https://www.udemy.com/course/learn-to-create-advance-ai-assistant-jarvis-20with-python/

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

https://pythonspot.com/personal-assistant-jarvis-in-python/

• Books Referred:

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

• Youtube Channels Referred:

- **Avi Upadhaya**
- **Code With Harry**

APPENDIX

```
import pyttsx3                          ##ALL THE LIRARIES
import speech_recognition as sr
import datetime
import wikipedia
import webbrowser
import random
import os
import pyautogui
import smtplib
import sys

from PyQt5 import QtWidgets, QtCore, QtGui    ##GUI LIBRARIES
from PyQt5.QtCore import QTimer, QTime, QDate, Qt
from PyQt5.QtGui import QMovie
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtWidgets import *
from PyQt5.uic import loadUiType
from JarvisGUI_2 import Ui_MainWindow
from sys import argv,exit

n = random.randint(0,23)
# print(n)

engine = pyttsx3.init('sapi5')            ##VOICE OUTPUT FNS
voices = engine.getProperty('voices')
# print(voices[1].id)
engine.setProperty('voice', voices[1].id)


def speak(audio):                         ##VOICE INPUT FNS
    engine.say(audio)
    engine.runAndWait()


def wishMe():                             ##WISHME FNS

    hour = int(datetime.datetime.now().hour)
    if hour>=0 and hour<12:
        speak("GOOD MORNING")

    elif hour>=12 and hour<18:
        speak("GOOD AFTERNOON")

    else:
        speak("GOOD EVENING")

    speak("I am Jarvis. Let me know how can I help you")


def sendEmail(to, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.ehlo()
    server.starttls()
    server.login('youremail@gmail.com', 'your-password')
    server.sendmail('youremail@gmail.com', to, content)
    server.close()

class MainThread(QThread):
    def __init__(self):
        super(MainThread, self).__init__()

    def run(self):
        self.TaskExecution()
            

    def takeCommand(self):              ##TAKE COMMAND FNS
        

        r = sr.Recognizer()
        with sr.Microphone() as source:
            print("Listening...")
            r.pause_threshold = 1
            audio = r.listen(source)

        try:
            print("Recognizing...")
            self.query = r.recognize_google(audio, language='en-in')
            print(f"User said: {self.query}\n")

        except Exception as e:
            
            print("Say that again please...")
            return "None"
        self.query = self.query.lower()
        return self.query
```

```
def TaskExecution(self):            ##TASK EXECUTION FNS
    wishMe()
    while True:

        self.query = self.takeCommand()

        if 'wikipedia' in self.query:
            speak('Searching Wikipedia...')
            self.query = self.query.replace("wikipedia", "")
            results = wikipedia.summary(self.query, sentences=2)
            speak("According to Wikipedia")
            print(results)
            speak(results)

        elif 'open youtube' in self.query:
            webbrowser.open("youtube.com")

        elif 'open google' in self.query:
            webbrowser.open("google.com")

        elif 'open stackoverflow' in self.query:
            webbrowser.open("stackoverflow.com")   ##"C:\Users\somay\Videos\MUSIX\AP-Gurinder-Shinda"

        elif 'play music' in self.query:
            music_dir = "C:\\Users\\somay\\Videos\\MUSIX\\AP-Gurinder-Shinda"
            songs = os.listdir(music_dir)
            os.startfile(os.path.join(music_dir, songs[n]))

        elif 'the time' in self.query:
            strTime = datetime.datetime.now().strftime("%H:%M:%S")
            speak(f"Sir, the time is {strTime}")

        elif 'open code' in self.query:
            codePath = "C:\\Program Files\\Microsoft VS Code\\Code.exe"
            os.startfile(codePath)

        elif 'screenshot' in self.query:
            image = pyautogui.screenshot()
            image.save('screenshot.png')
            speak('Screenshot taken.')
        elif 'email to keshav' in self.query:
            try:
                speak("What should I say?")
                content = self.takeCommand()
                to = "keshav.cse@acem.edu.in"
                sendEmail(to, content)
                speak("Email has been sent!")
            except Exception as e:
                print(e)
                speak("Sorry. email can't be send")

startExecution = MainThread()

class Main(QMainWindow):            ##OOPS BASED CMD FOR GUI
    def __init__(self):
        super().__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.ui.Startbutton.clicked.connect(self.StartTask)
        self.ui.Quit_button.clicked.connect(self.close)
        self.ui.YT_pushbutton.clicked.connect(self.youtube)
        self.ui.Chrome_pushbutton.clicked.connect(self.chrome)

    def youtube(self):
        webbrowser.open("youtube.com")

    def chrome(self):
        webbrowser.open("google.com")


    def StartTask(self):
        self.ui.movie = QtGui.QMovie("initial.gif")
        self.ui.Gif_1.setMovie(self.ui.movie)
        self.ui.movie.start()

        self.ui.movie = QtGui.QMovie("jarvis-iron-man.gif")
        self.ui.Gif_2.setMovie(self.ui.movie)
        self.ui.movie.start()

        self.ui.movie = QtGui.QMovie("d6a4db7983112c867f7ec4d71e754292.gif")
        self.ui.Gif_3.setMovie(self.ui.movie)
        self.ui.movie.start()
        startExecution.start()
```

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