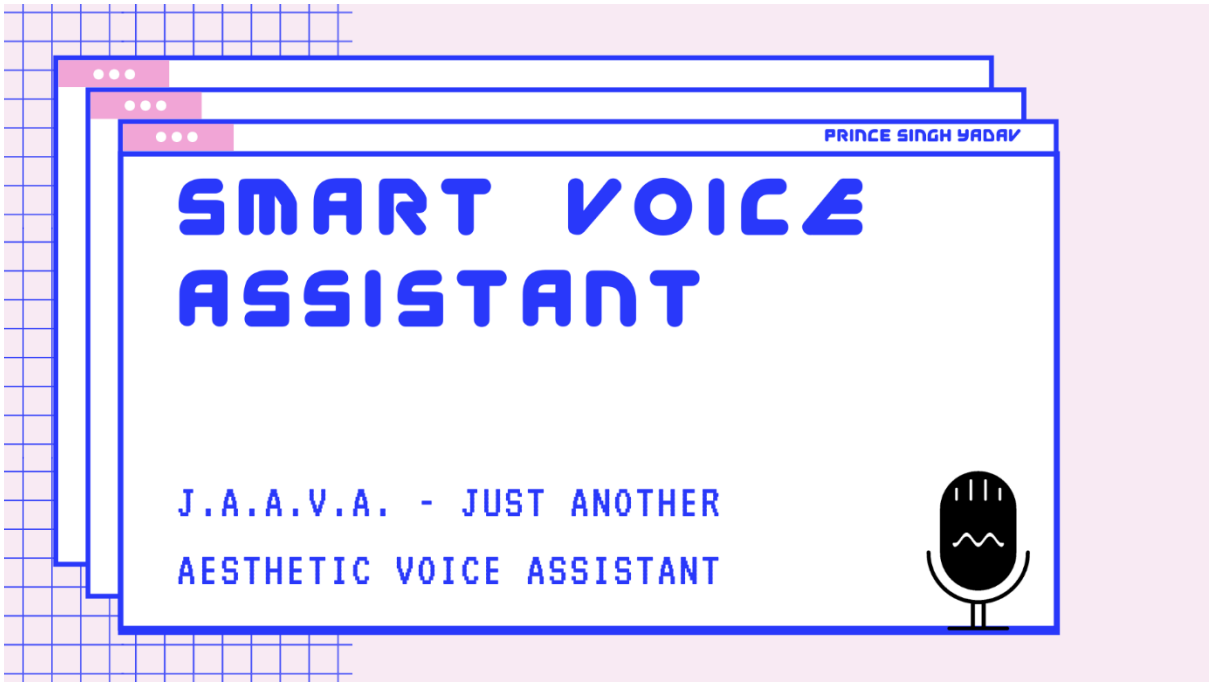
J.A.A.V.A

(Just Another Aesthetic Voice Assistant)

A Project Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Bachelor of Technology
in
Computer Science and Engineering

Prince Singh Yadav



Contents

Abstract of the Work: -

A Smart Assistant, also known as a virtual assistant is a form of software installed in a smart device such as a Smart Speaker or a Smart Phone that possesses the capability to perform tasks or services, and even possesses the necessary capabilities to answer questions that are asked by the user. They have recently become an essential link between the users and smart technology software and are proving essential for controlling smart plugs, smart thermostats and similar devices. By recognizing the huge opportunities and the possible advancements, we have chosen to develop our very own Smart Voice Assistant that makes life easy for its easy for this Joy of Engineering Project.

Keywords: - Smart Assistant, Smart Technology, Thermostats, Alexa, Siri

Introduction about the Smart Voice Assistant, J.A.A.V.A

The Smart Voice Assistant that we have developed J.A.A.V.A, is a voice assistant that is developed in such a way that it provides a lot of options for its particular user. It stands for **'Just Another Aesthetic Voice Assistant'**.

Our Assistant **J.A.A.V.A** comes with a lot of features like **Making Phone Calls**, **Scheduling Meetings** and **setting Appointments for the user**, **Gets Directions for the users**, **Sending Messages**, **Setting Reminders**, P**laying Music and various Videos** and can even **set alarms for users**. The Assistant can open any website in the browser, tell you the current weather or temperature

of a location, play you a song, and keep your news feed updated. The Voice Assistant can also answer questions for the users.

The thing about our Voice Assistant is that this Assistant speaks naturally with the user, grows along with the user and even entertains the user. By this Assistant we aim to give users an opportunity to organise their schedules and data in a much more effective way.

Popular Voice Assistants that we know and see people use include assistants like **Alexa, Siri** and **Google Assistant**.

Working and Designing of the Smart Voice Assistant

Our Smart Voice Assistant has been designed by using Python as its core Programming language and the various libraries available to us in it like: -

- **Speech Recognition**
- **Selenium (for web-based work from browser) and**
- **gTTS (for converting audio to text).**

The Assistant itself very user-friendly and does not require complicated commands to run and hence, can be used by people of all ages without any prior technical knowledge. This is very useful for such people.

The working of the Voice Assistant uses the following principles: -

Natural Language Processing (NLP): - This is used to understand the user's speech input.

Automatic Speech Recognition: - To understand the decipher the commands given in the form of the user's voice inputs.

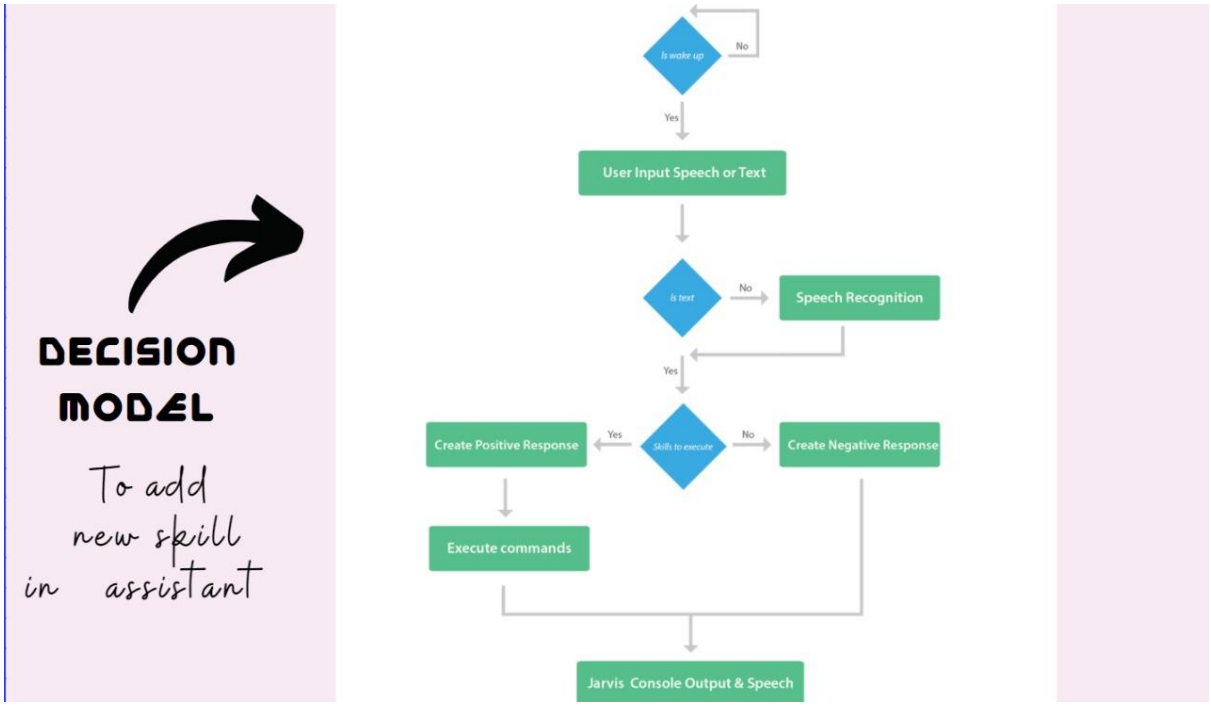
Inter Process Communication: - This is used to collect and gather important information from other software applications for usage.

We have also used technologies like **Speech-To-Text, Text Analyzing** and **Interpret Command.**

Speech to text: - This is a piece of software used to convert audio given in by the user to text.

Text Analyzing: - This is a piece of software that we can use for converting text to something that is understandable for computer.
Computer understands the commands of the user, so Virtual Assistants like **Siri** uses this to convert texts to computer command.

Interpret commands: - In this layer, that mapped computer command, go to server through internet and your speech is evaluated locally simultaneously.
A local recognizer communicates with server to judge whether command will be best handle locally or not.
Some common examples of Interpreting commands are: -
Play Music, Restaurant reservation etc.



Software and Hardware Requirements for the Assistant

Hardware Requirements: -

- A phone with a touch screen interface.

- The minimum Phone Ram should be of **512 MB**.
- The phone should be able to access the Internet. (Internet connectivity)
- The phone should have USB debugging mode for development and testing purposes

Software Requirements: -

- The Operating system of the device should be **Android 4.1/Windows 8.1/IOS 6** or a higher version.
- The kernel version of the device should be 3.0.16 or higher.
- The device must have the capability of Supporting other basic applications like Maps, Calendar, Camera, Web Connection etc.

ADVANTAGES AND DISADVANTAGES

Advantages: -

- These make small and smart hand-held devices to combine multiple features
- Allow you to export and import data
- Store various information
- Make to-do list
- Recognize face commands
- Controls various applications
- Provides services regarding locations
- Help to plan your whole day
- Reminds your important things on the accurate situation, location or time

Disadvantages: -

- Listening problem
    - VPA get problem to process wrong pronounced words
- Silent mode support
    - VPA gives response in voice output thus it doesn't work properly in silent mode
- Navigation languages
    - Most of the VPAs can understand only English language
- Internet access

- VPA needs internet connection to give desired output

THE CODE

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


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

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


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

def wishMe():
    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 JAAVA. how may I help you")

def takeCommand():
    #It takes microphone input from the user and returns string output
```

```
    r = sr.Recognizer()
    with sr.Microphone() as source:
        r.adjust_for_ambient_noise(source, 1.2)
        print("Listening...")
        r.pause_threshold = 1
        audio = r.listen(source)

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

    except Exception as e:
        # print(e)
        print("Say that again please...")
        return "None"
    return query

def sendEmail(to, content):
    server = smtplib.SMTP('smtp.gmail.com', 587)
```

```
    server.ehlo()
    server.starttls()
    server.login('mail id', 'pass')
    server.sendmail('mail id', to, content)
    server.close()

if _name_ == "_main_":
    wishMe()
    while True:
    # if 1:
        query = takeCommand().lower()

        # Logic for executing tasks based on query
        if 'wikipedia' in query:
            speak('Searching Wikipedia...')
            query = query.replace("wikipedia", "")
            results = wikipedia.summary(query, sentences=2)
            speak("According to Wikipedia")
            print(results)
            speak(results)
```

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

        elif 'open stack overflow' in query:
            webbrowser.open("stackoverflow.com")
        elif 'play music' in query:
            music_dir = " #if someone wants to play songs on the computer
            songs = os.listdir(music_dir)
            print(songs)
            os.startfile(os.path.join(music_dir, songs[0]))

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

        elif 'email' in query:
            try:
                speak("What should I say?")
                content = takeCommand()
```

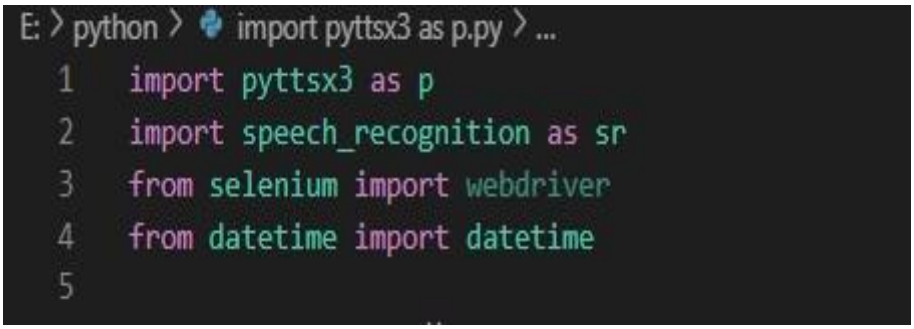
```
            to = "recipient mail"
            sendEmail(to, content)
            speak("Email has been sent!")
        except Exception as e:
            print(e)
            speak("I'm sorry the E mail wasn't sent successfully")

    elif 'exit code' in query:
        speak("Thank's for letting me serve you")
        sys.exit()

    elif 'playlist' in query:
        speak("now playing")
webbrowser.open('https://www.youtube.com/watch?v=tgY1b0AX7B8')

    elif  'amazon' in query:
        speak("what shoud i search for?")
        product = takeCommand()
        speak("fetching data for" + product)
        webbrowser.open('https://www.amazon.in/s?k=' + product + '&ref=nb_sb_noss_1')
```

SYNOPSIS OF THE PROJECT



- **Pyttsx3** is a library that will be used to convert "text-to-speech". By this we mean to say that whatever results we get, it will speak with the help of this library.
- Here we use the "**Speech recognition**" to help it recognize the voice of the user with the help of Google speech recognition.
- In this code of voice Assistant we have utilized a library available in python known as **Selenium**. This library allows us to do web-based work from browser. This means that the assistant can access the web browser for some tasks.
- By using the **datetime** feature the assistant can inform its user regarding the current data and current time when asked. Simply put it comes into play when the user wants results related to date and time

```
r = sr.Recognizer()
speak("Hello there, I am your personal voice assistant. How are you?")

with sr.Microphone() as source:

    r.energy_threshold=10000
    r.adjust_for_ambient_noise(source, 1.2)
    print("listening...")
    audio = r.listen(source)
    text = r.recognize_google(audio)
    print(text)
if "what" or "how" and "about" and "you" in text:
    speak("Just computing a few bits")
```

- Here we have used the **'r=sr.Recognizer()'** where the Recognizer class can help us while converting audio files into text. To access the Recognizer class, we can import its library by using

import speech_recognition as sr

- Here we also use our Microphone as the medium for giving our inputs with **sr.Microphone() as source.**

- And we set the **r.energy_threshold** value as **10,000**. We can think of the energy threshold as the loudness of the audio files. The values below the threshold are considered silence, and the values above the threshold are considered speech. This will improve the recognition of the speech when working with the audio file.
- Next for removing or for dulling the background noises we have used the **r.adjust_for_ambient_noise()** and we put its duration as 1.2. Now we can increase or decrease according to our needs. The more time we give it the better it does its work.

```
with sr.Microphone() as source:

    r.energy_threshold=10000
    r.adjust_for_ambient_noise(source, 1.2)
    print("listening...")
    audio = r.listen(source)
    text = r.recognize_google(audio)
    print(text)
```

- With this piece of code, JAAVA will listen to the source (Your device's microphone) as 'audio' and then recognize this audio using the "**recognize_google**" which performs speech recognition on audio using the Google Speech Recognition API and save that as text. Then it will print the recognized text so that user will get to know what JAAVA had recognized.

```
if "open" or "Open" and "google" or "Google" in text:
    speak("Opening google")
    class infow():
        def __init__(self):
            self.driver = webdriver.Chrome(executable_path='D:\chromedriver.exe')
elif "show" or "Show" or "current" or "Current" or "time" or "Time" in text:
    print("Current Time =", current_time)
    speak("the current time is:")
    speak(current_time)
elif "what" or "What" and "is" and "Date" or "date" and "Today" or "today" in text:
    print(current_date)
    speak("It is")
    speak(current_date)
elif "open" or "Open" and "calculator" or "Calculator" in text:
    subprocess.Popen('C:\\Windows\\System32\\calc.exe')

elif "open" or "Open" and "video" or "Video" and "Player" or "player" in text:
    subprocess.Popen('C:\Program Files\VideoLAN\VLC\vlc.exe')
```

- Now we have written some if-elif statements using which JAAVA will search out for certain keywords in the recognized text and accordingly perform the task. Like if JAAVA got something like “open google” in user’s recognized text then it will take the user to the google search page.
- If the query have keywords like “show current time” or “what is the date today”, it will do that accordingly. If user have asked JAAVA to open applications like calculator or video player and then it detects these keywords in the recognized text and open the required application for the user.

CONCLUSIONS

Virtual Personal Assistants are designed in such a way that they are very effective to organize your daily schedules, appointments, meetings etc. Now there are many Virtual Personal Assistants present in the market for usage on the various device platforms that are available.

These software Assistant applications are performing really well, even more than PDA Devices (Personal Device Assistant Devices) as they are being provided with all sorts of resources for our smartphones to make life easier and much more efficient for the user.

VPA’S (Virtual Personal Assistants) are also very much reliable than Human Personal Assistants because, VPA’s are portable and they can be used anytime. They also have a lot of Information than any assistants as they are connected to the Internet. This is very useful for the user and is also made very easy to access whenever required.