Bahria University

Karachi Campus

PROJECT REPORT

PROJECT TITLE: "ZIRA: Voice Assistant"

COURSE: ARTIFICIAL INTELLIGENCE LAB

TERM: FALL 2020, CLASS: BSE- 5(B)

Submitted By:

Syeda Bushra Naqvi	[57275]
Qaiser Abbas	[57245]

Submitted To:

Engr. Muhammad Rehan Baig

Signed: Remarks: Score:

Contents

PURPOSE:

This Software aims at developing a personal assistant for windows-based systems. The main purpose of the software is to perform the tasks of the user at certain commands, provided in speech. It will ease most of the work of the user as a complete task can be done on a single command. ZIRA draws its inspiration from Virtual assistants like Cortana for Windows and Siri for iOS. Users can interact with the assistant either through voice commands.

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 finger tips. 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 ecommerce 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 JIA. 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 ZIRA.

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 turning out 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.

- It can play music for you.
- It can do Wikipedia searches for you.
- It is capable of opening websites like Google, Youtube, etc., in a web browser.
- It is capable of opening your code editor or IDE with a single voice command.

PRODUCT GOALS AND 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.

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

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

IDE & LANGUAGE:

You can use any IDE but pycharm is recommended.

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 OOPs languages.

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 ZIRA, 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.

HARDWARE AND SOFTWARE 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
- Pycharm or anyother IDE

LIBRARIES:

- pip install pyttsx3
- pip install speechRecognition
- pip install Wikipedia

Pyttsx3:

Pyttsx stands for Python Text to Speech. It is a cross-platform Python wrapper for textto-speech synthesis. It is a Python package supporting common text-to-speech engines on Mac OS X, Windows, and Linux. It works for both Python2.x and 3.x versions. Its main advantage is that it works offline.

Sapi5:

Microsoft developed speech API.

Helps in synthesis and recognition of voice.

VoiceID:

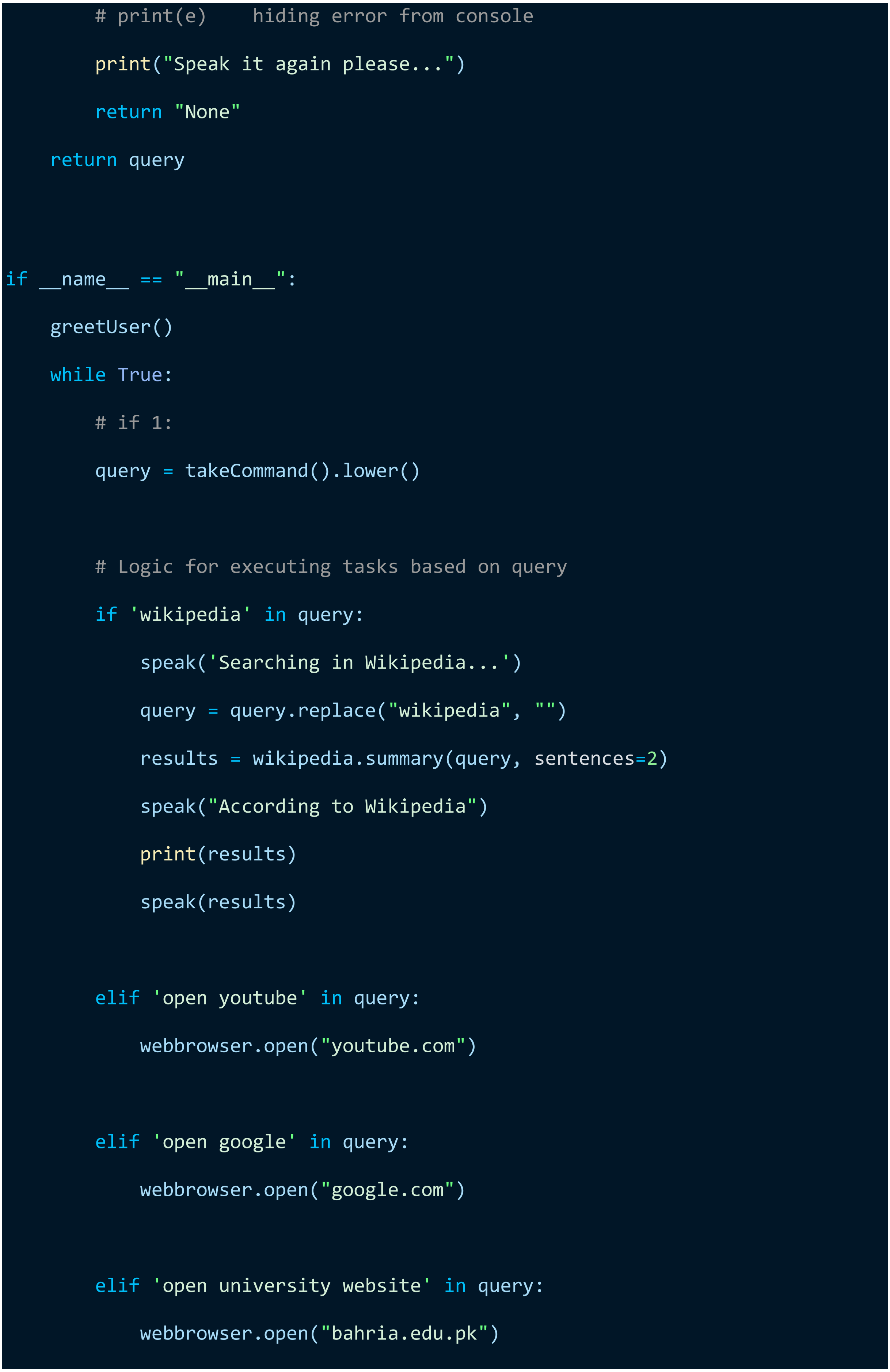
Voice id helps us to select different voices.

voice[0].id = Male voice

voice[1].id = Female voice

COMPLETE CODE:

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

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


def speak(audio):
    """
This method will allow Zira to speak, It take our voice as an argument
    """
    engine.say(audio)
    engine.runAndWait()


def greetUser():
    """
This method will always greet the user in start.
    """
    hour = int(datetime.datetime.now().hour)
    if hour >= 0 and hour < 11:
        speak("Good Morning!")
    elif hour >= 11 and hour < 18:
        speak("Good Afternoon!")

    else:
        speak("Good Evening!")

    speak("My name is Zira, I am your voice assistant. Please tell me how may I help you?")


def takeCommand():
    """
    It takes microphone input from the user and returns string output, return None in case of any problem
    """

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("how can I help?")
        r.pause_threshold = 1
        audio = r.listen(source)

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

    except Exception as e:
```



```
        # print(e)    hiding error from console
        print("Speak it again please...")
        return "None"
    return query


if __name__ == "__main__":
    greetUser()
    while True:
        # if 1:
        query = takeCommand().lower()

        # Logic for executing tasks based on query
        if 'wikipedia' in query:
            speak('Searching in 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("google.com")

        elif 'open university website' in query:
            webbrowser.open("bahria.edu.pk")
        elif 'open stack overflow' or 'open stackoverflow' in query:
            speak("Here you go to Stack Over flow Happy coding")
            webbrowser.open("stackoverflow.com")

        elif 'play music' in query:
            music_dir = 'F:\\MP3'
            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 'open chrome' in query:
            codePath = r"C:\Program Files\Google\Chrome\Application\chrome.exe"
            os.startfile(codePath)

        elif 'open git' in query:
            codePath = "C:\\Program Files\\Git\\git-bash.exe"
            os.startfile(codePath)

        elif 'quit' or 'shut' in query:
            speak("Thanks for giving me your time")
            exit()
```

CONCLUSION:

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 time. In this respect, the ability of personal assistants to accurately recognize spoken words is a prerequisite for them to be adopted by consumers.

Through ZIRA 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, playing music, opening chrome and time 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 entire code along with some additional files for this voice assistant is located in our git repo.

https://github.com/iQaiserAbbas/Voice-Assistant-in-Python

REFERENCES:

Websites referred

- www.stackoverflow.com
- www.pythonprogramming.net
- www.codecademy.com
- www.tutorialspoint.com

Books referred

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

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