VIRTUAL ASSISTANT

A mini project report submitted by

JOSWIN V JAISON(URK18CS097)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

under the supervision of

Dr. R.VENKATESAN, Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed-to-be-under Sec-3 of the UGC Act, 1956)

Karunya Nagar, Coimbatore - 641 114. INDIA

March 2021

Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)
A CHRISTIAN MINORITY RESIDENTIAL INSTITUTION
AICTE Approved & NAAC Accredited

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled, "**Virtual Assistant**" is a bonafide record of Mini Project work done during the even semester of the academic year 2020-2021 by

Joswin V Jaison (Reg. No: URK18cs097)

in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Karunya Institute of Technology and Sciences.

Submitted for the Viva Voce held on _____29/2/2021_____

Project Coordinator **Signature of the Guide**

ACKNOWLEDGEMENT

First and foremost, I praise and thank ALMIGTHY GOD whose blessings have bestowed in me the will power and confidence to carry out my project.

I am grateful to our beloved founders Late**. Dr. D.G.S. Dhinakaran, C.A.I.I.B, Ph.D** and **Dr. Paul Dhinakaran, M.B.A, Ph.D**, for their love and always remembering us in their prayers.

I extend my thanks to our Vice Chancellor **Dr. P. Mannar Jawahar, Ph.D** and our Registrar **Dr. Elijah Blessing, M.E., Ph.D,** for giving me this opportunity to do the project.

I would like to thank **Dr. Prince Arulraj, M.E., Ph.D.,** Dean, School of Engineering and Technology for his direction and invaluable support to complete the same.

I would like to place my heart-felt thanks and gratitude to **Dr. J. Immanuel John Raja, M.E., Ph.D.,** Head of the Department, Computer Science and Engineering for his encouragement and guidance.

I feel it is a pleasure to be indebted to, **Mr. J. Andrew, M.E, (Ph.D.),** Assistant Professor, Department of Computer Science and Engineering and **Dr.R.Venkatesan (guide)** for their invaluable support, advice and encouragement.

I also thank all the staff members of the Department for extending their helping hands to make this project a successful one.

I would also like to thank all my friends and my parents who have prayed and helped me during the project work.

ABSTRACT:

The project aims to develop a personal assistant for windows operating system.I have drawn the inspiration from virtual assistants like Cortana and Siri.It has been designed to provide a user friendly interface for carrying a variety of tasks by employing certain well defined commands. Users can interact with the system through voice commands. As a personal assistant it will assist the user to perform various tasks like general conversation, get updated news, get the weather and temperature of a particular place, play songs from the user's playlist ,open any websites, searching queries in google, remind the user about birthdays or any important dates, get the exact time, read content in multiple languages including German, French as well as Indian languages like Malayalam, Tamil, Telugu, Kannada, Marathi. It will also perform mathematical and geographical computations. It will read the content from Wikipedia. It will enable the user to send Email if less secure apps is enabled on Gmail account. The user statements are analyzed with the help of Artificial Intelligence and most importantly python modules. It was developed in accordance with python modules likepyttsx3, speech recognition, datetime, Wikipedia, webbrowser, os,smtlib, pandas, io, requests, time, gttS, wolframalpa

Keywords

Artificial Intelligence,Python,Virtual Assistant,Desktop assistant,Personal assistant,Speech recognition,Pyttsx3.

CONTENTS

Acknowledgement 3

Abstract 4

1. Introduction **5**

1.1 Introduction 5

1.2 Objectives 6

1.3 Motivation 7

1.4 Overview of the Project 8

2. Analysis and Design

2.1 Functional Requirements 8

2.2 Non-Functional Requirements 9

2.3 Use case diagram 10

2.4 Sequence diagram 12

3. Implementation . 13

3.1. Modules Description 13

3.2. Implementation Details 18

3.3. Tools used 25

4. Test results/experiments/verification . 26

4.1. Testing 26

4.2. Results 27

4.3. Verification

5. Conclusions and Further Scope **32**

6.References **34**

1. INTRODUCTION

1.1 INTRODUCTION

A Virtual assistant also called an AI assistant or digital assistant is an application that understands natural language voice commands and complete tasks for the user. Such tasks, historically performed by a personal assistant or secretary, include taking dictation, reading text or email messages aloud, looking up phone numbers, scheduling, placing phone calls and reminding the end user about appointments. Popular virtual assistants currently include Amazon Alexa, Apple's Siri, Google Assistant and Microsoft's Cortana -- the digital assistant built into Windows Phone 8.1 and Windows 10. Though this definition focuses on the digital form of virtual assistants, the term virtual assistant, or virtual personal assistant, is also commonly used to describe contract workers who work from home doing administrative tasks typically performed by executive assistants or secretaries. Virtual assistants can also be contrasted with another type of consumer-facing AI programming, called smart advisers. Smart adviser programs are subject-oriented, while virtual assistants are task-oriented. Virtual assistants are typically cloud-based programs that require internet-connected devices and/or applications to work. Three such applications are

Siri on Apple devices, Cortana on Microsoft Devices and Google Assistant on Android devices.

1.2OBJECTIVE

The major objective of this project is to provide the user with various functionalities to complete their task easily by using the virtual assistant.This will not only aid them in performing tasks like searching the web,playing the music,extracting the web data and weather information.But also helps them in performing their daily operations like sending Email, reminding important dates and activities to do and also to perform geographical and mathematical computations.Another core function performed by this virtual assistant is that it can read a text file in multiple languages so it will be useful for individuals learning languages .It will also benefit users who cannot read a particular language but can understand to study the contents written in that particular language.

1.3.MOTIVATION

We are all aware of Cortanana,Siri,Google assistant and many other virtual assistants which are designed to the task of users in windows,android and ios platforms.So I try to design a virtual assistant using python which help me as my desktop assistant and assist me in performing daily tasks. The main purpose of the software is to perform the tasks of the user at certain user instructions or

commands, provided in the mode of speech as well as text. It will ease most of the work of the user as a complete task can be done on a single command. Jarvis

draws its inspiration from Virtual assistants like Cortana for Windows and Siri

for iOS.It will enable the user to interact through speech.The programming language used is python.

1.4.OVERVIEW OF THE PROJECT

As a personal assistant, Jarvis assists the end-user with day-to-day activities like general human conversation, searching queries in various search engines,live weather conditions, read text in multiple languages, retrieve current affairs,play song from the playlist,sending emails,perform calculations and Wikipedia searches etc.The virtual assistant begins by greeting the user based on time as it will say good morning when it is morning and good afternoon when it is afternoon respectively. We may also see any to-do works as we are running the program. The user statements are analysed with the help of python modules and artificial intelligence.

2.ANALYSIS AND DESIGN:

2.1FUNCTIONAL REQUIREMENTS

Functional requirements are product features or functions that developers must implement to enable users to accomplish their tasks. So, it's important to make them clear both for the development team and the stakeholders.In this project these functional requirements are required.

*The virtual assistant should greet the user respective to the time and should provide a remainder of to-do activities including birthday reminders.

*The virtual assistant should respond to the user when a specific instruction is given.

*It should automatically print ‘listening’ for the user to speak and ‘recognizing’ after the user give instruction.

*Execute the command ‘say that again please’ if it couldn’t recognize the instruction spoke by the user.

*When sending email using virtual assistant the user should disable his less secure apps feature in Gmail.

*An enhanced interactive python is required for the program to run.

*The text printed inside the speak function need to be executed by the virtual assistant**.**

2.2NON FUNCTIONAL REQUIREMENTS

A non-functional requirement defines the performance attribute of a software system. Types of Non-functional requirement are Scalability Capacity, Availability, Reliability, Recoverability, Data Integrity, etc. Example of Non Functional Requirement is Employees never allowed to update their salary information.In this project these non functional requirements need to be satisfied.

*The program should halt after 6 seconds after the news is displayed.

*If time is between 12 am to 12pm the assistant should greet user with good morning.

*If time is between 12 pm and 6 pm the assistant should greet with good afternoon.

*If the time is after 6pm and before 12 am then the user should be acknowledged with good evening.

Architecture

The program is executed in a free and open source scientific environment called spyder written in python.Various python modules are used in this project.

*Pyttsx3 module: A python library that will help us to convert text to speech. In short, it is a text-to-speech library. It works offline, and it is compatible with Python 2 as well as Python 3.

*sapi5: A Microsoft developed speech API. Helps in synthesis and recognition of voice.

*Voice Id:voice Id help us to select different voices.

voice[0].id = Male voice

voice[1].id = Female voice

*Writing speak function[speak()] :convert text to speech and without this command the speech will not be audible to us. It includes engine.say(audio) and engine.runandwait().

Nonfunctional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs They ensure the usability and

effectiveness of the entire system. Broadly, functional requirements define what a system is supposed to do and non-functional requirements define how a system is supposed to be. Functional requirements are usually in the form of "system shall do <requirement>", an individual action or part of the system, perhaps explicitly in the sense of a mathematical function, a black box description input, output, process and control functional model or IPO Model. In contrast, non-functional requirements are in the form of "system shall be <requirement>", an overall property of the system as a whole or of a particular aspect and not a specific function. The system's overall properties commonly mark the difference between whether the development project has succeeded or failed.

2.3.USE CASE DIAGRAM

A use case diagram is usually simple. It does not show the detail of the use cases: It only summarizes some of the relationships between use cases, actors, and systems. It does not show the order in which steps are performed to achieve the goals of each use case. As said, a use case diagram should be simple and contains only a few shapes. If yours contain more than 20 use cases, you are probably misusing use case diagram. There are many different UML diagrams that serve different purposes (as you can see from the UML diagram tree above). You can describe those details in other UML diagram types and documents, and have them be linked from use cases. Use cases represent only the functional requirements of a system. Other requirements such as business rules, quality of service requirements, and implementation constraints must be represented separately, again, with other UML diagrams. Use case diagrams are typically developed in the early stage of development and people often apply use case modeling for the following purposes:

Specify the context of a system

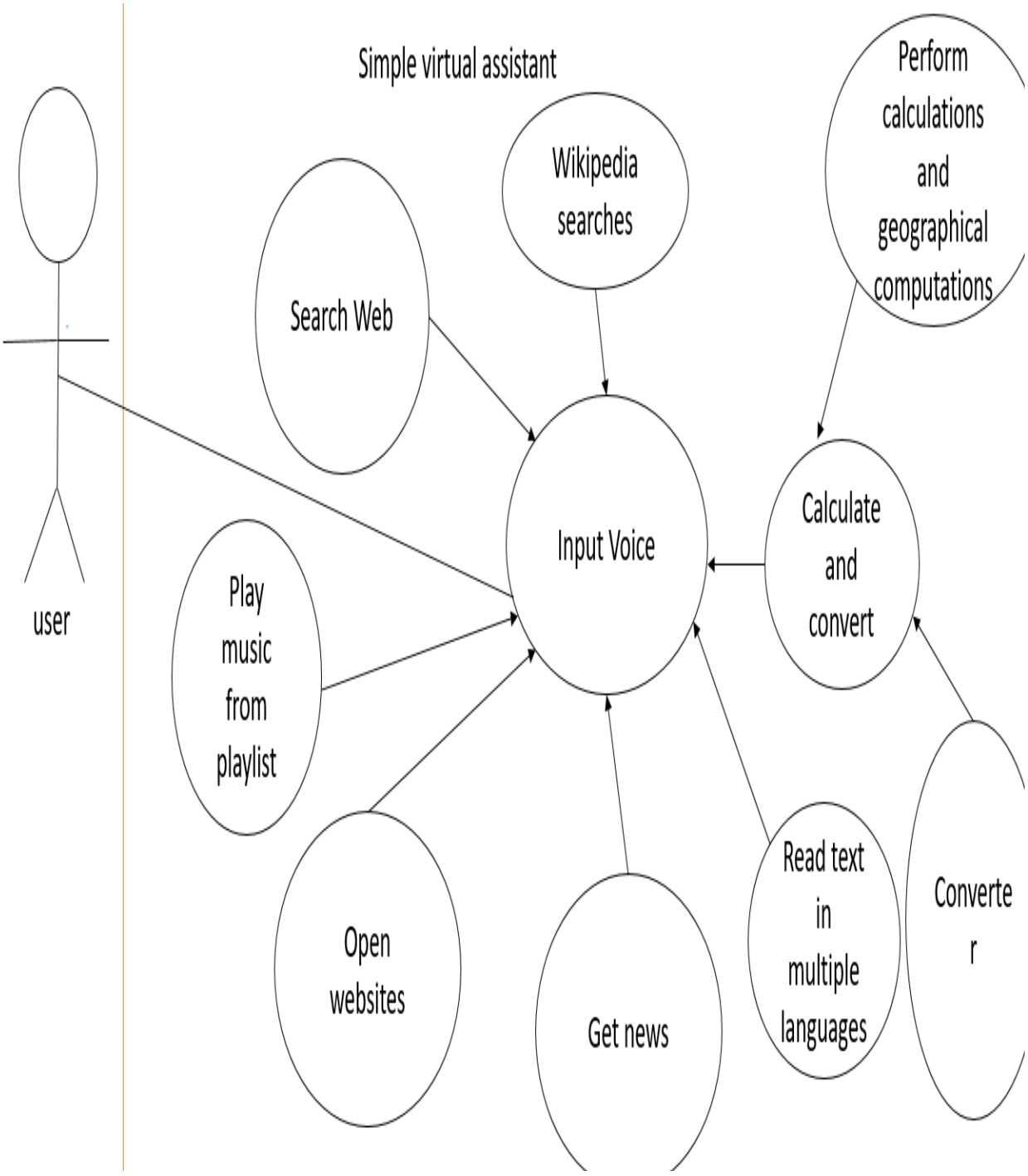
Capture the requirements of a system

Validate a systems architecture

Drive implementation and generate test cases

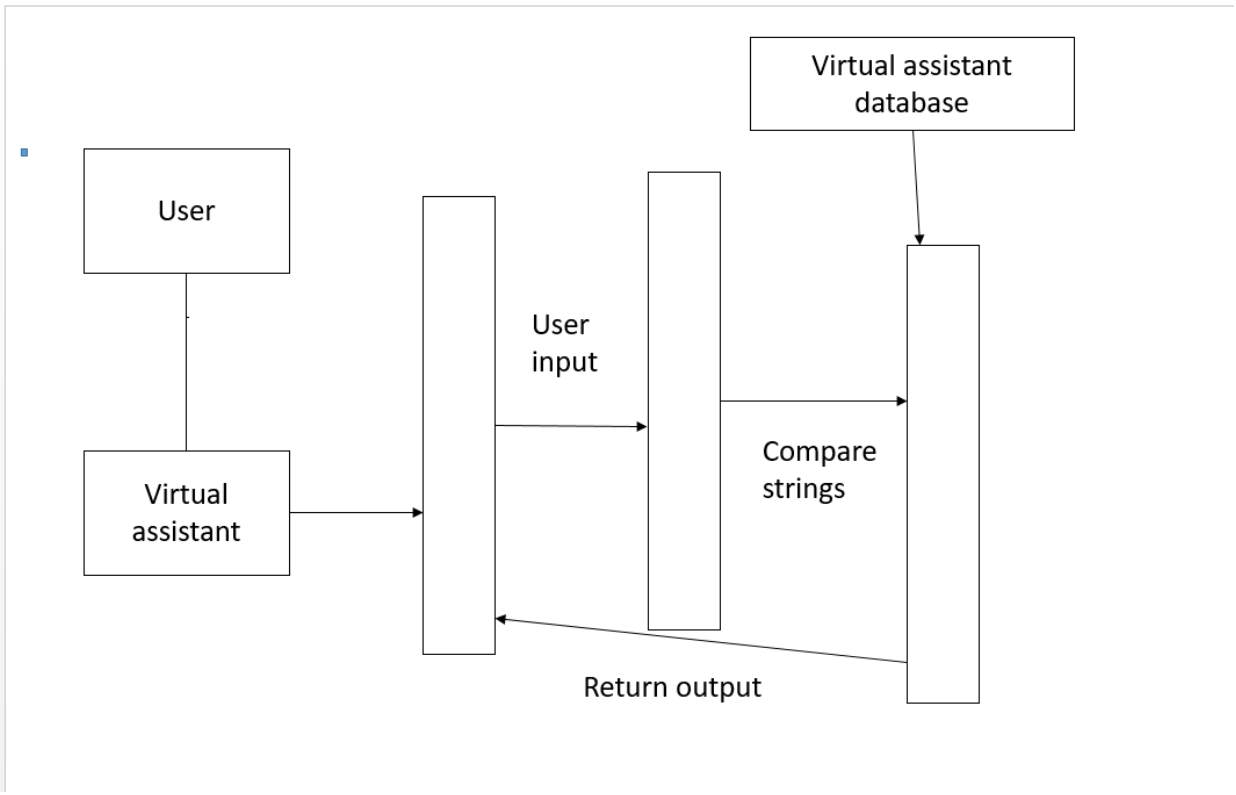
Developed by analysts together with domain experts

The use case diagram of the Virtual Assistant is given below :



Use case diagram

2.4 SEQUENCE DIAGRAM



3 .IMPLEMENTATION

3.1MODULES DESCRIPTION

Pyttsx3: pyttsx3 is a text-to-speech conversion library in Python. ... it is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is female and the second is male which is provided by "sapi5" for windows.

Speech Recognition: Speech recognition is the process of converting spoken words to text. Python supports many speech recognition engines and APIs, including Google Speech Engine, Google Cloud Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text. First, make sure you have all the requirements listed in the "Requirements" section.

The easiest way to install this is using `pip install SpeechRecognition`.

Otherwise, download the source distribution from PyPI, and extract the archive.

In the folder, run `python setup.py install`.

Datetime: The datetime module supplies classes for manipulating dates and times.

Date and time objects may be categorized as "aware" or "naive" depending on whether or not they include timezone information.

With sufficient knowledge of applicable algorithmic and political time adjustments, such as time zone and daylight saving time information, an aware object can locate itself relative to other aware objects. An aware object represents a specific moment in time that is not open to interpretation. 1

A naive object does not contain enough information to unambiguously locate itself relative to other date/time objects. Whether a naive object represents Coordinated Universal Time (UTC), local

time, or time in some other timezone is purely up to the program, just like it is up to the program whether a particular number represents metres, miles, or mass. Naive objects are easy to understand and to work with, at the cost of ignoring some aspects of reality.

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 MediaWiki API so you can focus on using Wikipedia data, not getting it. **Wikipedia** is a multilingual online encyclopedia created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system. In order to extract data from Wikipedia, we must first install the Python Wikipedia library, which wraps the official Wikipedia API. This can be done by entering the command below in your command prompt or terminal:

Pip install Wikipedia

Web Browser: The webbrowser module provides a high-level interface to allow displaying Web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.

Under Unix, graphical browsers are preferred under X11, but text-mode browsers will be used if graphical browsers are not available or an X11 display isn't available. If text-mode browsers are used, the calling process will block until the user exits the browser. If the environment variable BROWSER exists, it is interpreted as the os.pathsep-separated list of browsers to try ahead of the platform defaults. When the value of a list part contains the string %s, then it is interpreted as a literal browser command line to be used with the argument URL substituted for %s; if the part does not contain %s, it is simply interpreted as the name of the browser to launch.

In Python, webbrowser module provides a high-level interface which allows displaying Web-based documents to users. The webbrowser module can be used to launch a browser in a platform-independent manner as shown below:

import webbrowser

```
webbrowser.open('http://www.python.org')
```

Os: The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system. Consider Current Working Directory(CWD) as a folder, where the Python is operating. Whenever the files are called only by their name, Python assumes that it starts in the CWD which means that name-only reference will be successful only if the file is in the Python's CWD.

Smtlib: The smtplib module defi nes an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. For details of SMTP and ESMTP operation, consult RFC 821 (Simple Mail Transfer Protocol) and RFC 1869 (SMTP Service Extensions). comes with the built-in smtplib module for sending emails using the Simple Mail Transfer Protocol (SMTP). smtplib uses the RFC 821 protocol for SMTP. The examples in this tutorial will use the Gmail SMTP server to send emails, but the same principles apply to other email services. Although the majority of email providers use the same connection ports as the ones in this tutorial, you can run a quick Google search to confirm yours.

Pandas: In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The *pandas* package is the most important tool at the disposal of Data Scientists and Analysts working in Python today. The powerful machine learning and glamorous visualization tools may get all the attention, but pandas is the backbone of most data projects. Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel. Pandas allows various data manipulation operations such as merging,[9] reshaping,selecting, as well as data cleaning, and data wrangling features.

IO: This module is a part of the standard library, so there's no need to install it separately using pip. To import the io module, we can do the following:

```
import io
```

In the io module there are 2 common classes which are very useful for us:

BytesIO -> I/O operations on byte data

StringIO -> I/O operations on string data

We can access these classes using io.BytesIO and io.StringIO.

Requests: The requests module allows you to send HTTP requests using Python. The HTTP request returns a Response Object with all the response data (content, encoding, status, etc).

Time: Python has a module named time to handle time-related tasks. To use functions defined in the module, we need to import the module first. Here's how:

```
Import time
```

Here are commonly used time-related functions. The time() function returns the number of seconds passed since epoch. The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code. One of the ways you can manage the concept of Python time in your application is by using a floating point number that represents the number of seconds that have passed since the beginning of an era—that is, since a certain starting point.

WolframAlpha: The Wolfram|Alpha Webservice API provides a web-based API allowing the computational and presentation capabilities of Wolfram|Alpha to be integrated into web, mobile, desktop, and enterprise applications. Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledgebase and AI technology. It is made possible by

the Wolfram Language. Wolfram Alpha is a computational search engine that tends to evaluate what the user asks. ... Wolfram Alpha will be able to evaluate the question and respond with an answer like “15 degrees centigrade” or “Donald Trump”. Wikipedia, however, is a search engine that unlike Wolfram, does not compute or evaluate the question but rather searches for the keywords in the query. For example, Wikipedia cannot answer the questions like “What is the current weather in London” or “Who is the president of United State of America” but can search for keywords like “Donald Trump” or “London”.

Gtts: There are several APIs available to convert text to speech in Python. One of such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file.The gTTS API supports several languages including English, Hindi, Tamil, French, German and many more. The speech can be delivered in any one of the two available audio speeds, fast or slow. However, as of the latest update, it is not possible to change the voice of the generated audio. Google text to speech is a life saver when it comes to converting text to speech. To install gTTS and to use it, we need to type the below command.

Pip install gtts

Playsound: The playsound module contains only a single function named playsound().It requires one argument: the path to the file with the sound we have to play. It can be a local file, or a URL.There’s an optional second argument, block, which is set to True by default. We can set it to False for making the function run asynchronously.It works with both WAV and MP3 files. Play sound on Python is easy. There are several modules that can play a sound file (.wav). These solutions are cross platform (Windows, Mac, Linux). The main difference is in the ease of use and supported file formats. All of them should work with Python 3. The audio file should be in the same directory as your python program, unless you specify a path. The playsound module is a cross platform module that can play audio files. This doesn’t have any dependencies, simply install with pip in your virtualenv and run You can play sound files with the pydub module. It’s available in the pypi repository (install with pip). This module can use PyAudio and ffmpeg underneath. The module snack sound kit can play several audio files: WAV, AU, AIFF, MP3, CSL, SD, SMP, and NIST/Sphere. We can install it with your package manager: ‘apt install python3-tksnack’. For old versions there’s ‘python-tksnack’. This module depends on Tkinter. That means that to play sound

with this module, you'd also have to import the gui module Tkinter. The module doesn't seem to have been updated in a while. You can also play sounds natively on your system. This requires you to have some kind of audio player installed on the terminal. On Linux you can use mpg123 for that. This simply plays the mp3 file with an external player.

3.2 IMPLEMENTATION DETAILS

Defining Speak Function:

The first and foremost thing for an A.I. assistant is that it should be able to speak. To make our J.A.R.V.I.S. talk, we will make a function called speak(). This function will take audio as an argument, and then it will pronounce it.

*def speak(audio):

    pass

Now, the next thing we need is audio. We must supply audio so that we can pronounce it using the speak() function we made. We are going to install a module called pyttsx3.Pyttsx3 is a python library that will help us to convert text to speech.In short it is a text to speech library.It works offline and it is compatible with python 2 as well as python 3.Sometimes there is a chance that these errors may occur.

*No module named win32com. client.

*No module named win32.

*No module named win32api.

Then we need to install pipwin using the following command:

*Pip install pipwin32

After successfully installing pyttsx3, import this module into your program.

*import pyttsx3

Engine=pyttsx3.init(‘sapi5’)

Voice=Engine.getProperty(‘voices’)

Sapi5 is Microsoft developed speech API.It helps in the recognition and synthesis of voice.Voice id help us to select different voices.Voice[0].id means it is male voice.Voice[1].id means it is female voice.

Writing speak() function:

It help us to convert text to speech.

*def speak(audio):

    engine.say(audio)

    engine.runAndWait()

Creating main() function:

We will create a main() function, and inside this main() Function, we will call our speak function.Whatever we speak inside the speak() function it will be converted to speech.

Creating wishme() function:

Wish Me function greet the user based on the time of current user or pc.To provide current or living time to A.I we need to import a module called datetime.

*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 your virtual assistant. Please tell me how may I help you")
```

Defining take command function:

The next most important thing for our AI assistant is that it shoud take the command with the help of microphone of the users system.Now we will make a takeCommand() function. With the help of takeCommand() function our A.I assistant will return a string output by taking microphone input from the user.Before defining a command() function we need to install a module called SpeechRecognition.Install this module by:

*Pip install speechRecognition

After successfully installing this module import this module into program by writing an import statement.

*import speechRecognition as sr

Let's start coding the takeCommand() function :

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

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
```

```
        r.pause_threshold = 1
        audio = r.listen(source)
```

We have successfully created our takeCommand() function. Now we are going to add a try and except block to our program to handle errors effectively.

```
try:
        print("Recognizing...")
        query = r.recognize_google(audio, language='en-in') #Using google for voice recognition.
        print(f"User said: {query}\n")  #User query will be printed.

    except Exception as e:
        # print(e)
        print("Say that again please...")   #Say that again will be printed in case of improper voice
        return "None" #None string will be returned
    return query
```

To search something on Wikipedia:

We need to install the Wikipedia module to the program.We can type pip install Wikipedia to install this module. After successfully installing the Wikipedia module, import it into the program by writing an import statement.

```
if __name__ == "__main__":
    wishMe()
    while True:
```

```
# if 1:
    query = takeCommand().lower() #Converting user query into lower case

    # Logic for executing tasks based on query
    if 'wikipedia' in query:  #if wikipedia found in the query then this block will be executed
        speak('Searching Wikipedia...')
        query = query.replace("wikipedia", "")
        results = wikipedia.summary(query, sentences=2)
        speak("According to Wikipedia")
        print(results)
        speak(results)
```

In the above code, we have used an if statement to check whether Wikipedia is in the search query of the user or not. If Wikipedia is found in the user's search query, then two sentences from the summary of the Wikipedia page will be converted to speech with the speak function's help.

To open youtube site in a web browser:

To open any website, we need to import a module called webbrowser. It is an in-built module, and we do not need to install it with a pip statement; we can directly import it into our program by writing an import statement.

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

Here, we are using the elif loop to check whether Youtube is in the user's query. Let' suppose the user gives a command as "J.A.R.V.I.S., open youtube." So, open youtube will be in the user's query, and the elif condition will be true.

To open google site in a web browser:

The logic used to open youtube is used to open google site in a web browser.

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

Defining task to play music:

To play music, we need to import a module called os. Import this module directly with an import statement.

```
elif 'play music' in query:
        music_dir = 'D:\\Non Critical\\songs\\Favorite Songs2'
        songs = os.listdir(music_dir)
        print(songs)
        os.startfile(os.path.join(music_dir, songs[0]))
```

n the above code, we first opened our music directory and then listed all the songs present in the directory with the os module's help. With the help of os.startfile, we can play any song of our choice. I am playing the first song in the directory. However, we can also play a random song with the help of a random module. Every time we command to play music, A.I assistant will play any random song from the song directory.

Defining task to know current time:

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

In the above, code with using datetime() function and storing the current or live system into a variable called strTime. After storing the time in strTime, we are passing this variable as an argument in speak function. Now, the time string will be converted into speech.

Defining task to open any application:

Here I have written code to open VS code.

```
elif 'open code' in query:
        codePath = "C:\\Users\\Joswin\\AppData\\Local\\Programs\\Microsoft VS Code\\Code.exe"
        os.startfile(codePath)
```

To open the VS Code or any other application, we need the code path of the application.

Steps to get the code path of the application:

Step 1: Open the file location.

Step 2: Right-click on the application and click on properties.

Step 3: Copy the target from the target section.

After copying the target of the application, save the target into a variable. Here, I am saving the target into a variable called codePath, and then we are using the os module to open the application.

Defining task to send mail

To send an email, we need to import a module called smtplib. Simple Mail Transfer Protocol (SMTP) is a protocol that

allows us to send emails and to route emails between mail servers. An instance method called sendmail is present in the SMTP module. This instance method allows us to send an email. It

takes 3 parameters:The sender,receiver and the message. We will create a sendEmail() function, which will help us send emails to one or more than one recipient.

```
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()
```

We need to enable the less secure apps in our Gmail.

Defining task to read the text in multiple languages:

TTS gives access to your content to a greater population (with literacy difficulties, learning disabilities, reduced vision and those learning a language). With the help of this technology, we can learn the pronunciation of any language. Text-to-speech (TTS) technology reads words on computers, smartphones etc... (including Word, document, online web pages) and convert them into audio. A TTS Engine converts written text to a phonemic representation (Sounds of a word), then converts the phonemic representation to waveforms (WaveNet is a deep neural network for generating raw audio) that can be output as sound. Google Text-to-Speech is a python interface for google Text-to-Speech API. gTTS powers applications to read aloud (speak) the text on the screen with support for many languages (English , Hindi, Tamil etc...).

3.3 TOOLS USED

1.Spyder: Spyder, the Scientific Python Development Environment, is a free integrated development environment (IDE) that is included with Anaconda. It includes editing, interactive testing, debugging, and introspection features. It includes editing, interactive testing, debugging, and introspection features.

2.Anaconda: Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

3.M.S-Excel: MS Excel is a commonly used Microsoft Office application. It is a spreadsheet program which is used to save and analyse numerical data. A spreadsheet is in the form of a table comprising rows and columns. The rectangular box at the intersection point between rows and columns forms a cell.

4.VLC Media Player: VLC media player is a free and open-source, portable, cross-platform media player software, and streaming media server developed by the VideoLAN project. VLC is available for desktop operating systems, and mobile platforms, such as Android, iOS, iPadOS, Tizen, Windows 10 Mobile, and Windows Phone

5.Google Colab: Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

6.Jupyter Notebook: Project Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages". It was spun off from IPython in 2014 by Fernando Pérez.

7.Python 3.0: Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

8.Notepad: Notepad is a simple text editor for Microsoft Windows and a basic text-editing program which enables computer users to create documents. It was first released as a mouse-

based MS-DOS program in 1983, and has been included in all versions of Microsoft Windows since Windows 1.0 in 1985.

9.Selenium: Selenium is a portable framework for testing web applications. Selenium provides a playback tool for authoring functional tests without the need to learn a test scripting language

4.TEST RESULTS/EXPERIMENT VERIFICATION

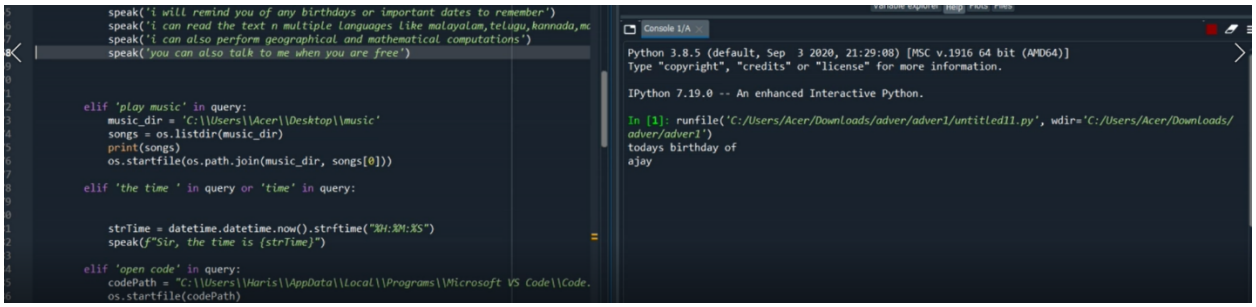
4.1 TESTING

The AI assistant created is tested with the help of spyder package in anaconda navigator.Errors occurred is solved with the help of stack overflow community. Stack Overflow is a question and answer site for professional and enthusiast programmers. It is a privately held website, the flagship site of the Stack Exchange Network, created in 2008 by Jeff Atwood and Joel Spolsky. It features questions and answers on a wide range of topics in computer programming.A working computer with medium processor capacity is required for the program to run.We can launch spyder from command prompt or from anaconda navigator.During testing the application is executed with different users whose frequency in voice as well as speaking mode is different.It was found that the AI assistant was able to answer each of the questions from user.The most efficient outcome will be produced from the process of continuous training.

So when testing we will get an accurate result.If the Virtual assistant is unable to recognize it,it will give results according to that.

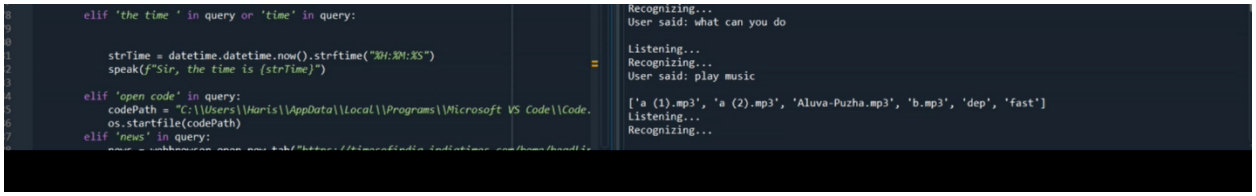
4.2 RESULTS AND VERIFICATION

a.Birthday reminder

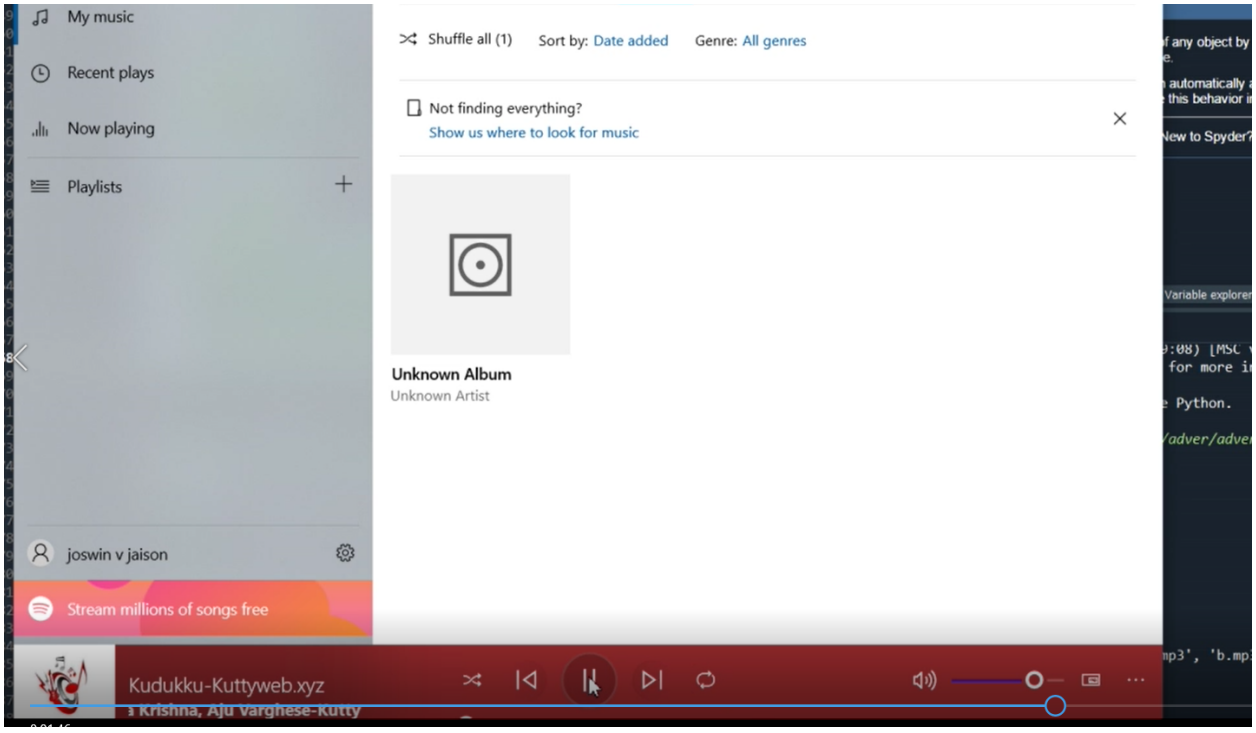


Birthday reminder when opening the Assistant as well as welcome the user with greetings.

b.Play music from the user's playlist randomly.

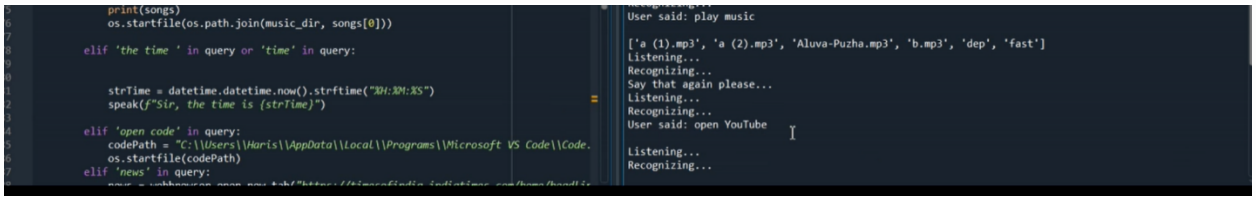


User said 'play music'

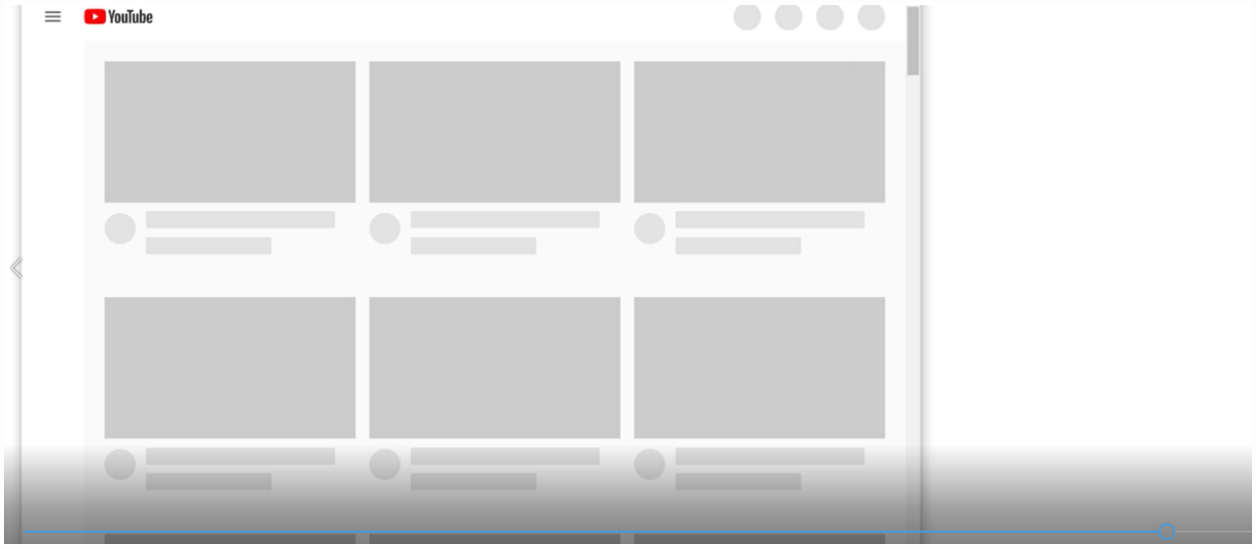


The AI assistant open music player based on command by the user.

c.Open youtube application

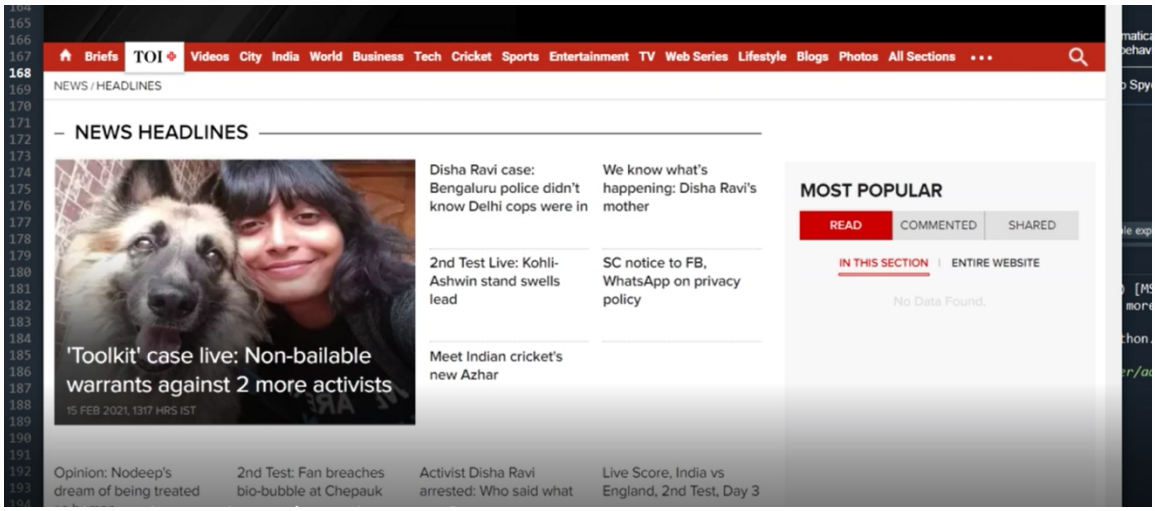


The user ask the user to ‘open Youtube’



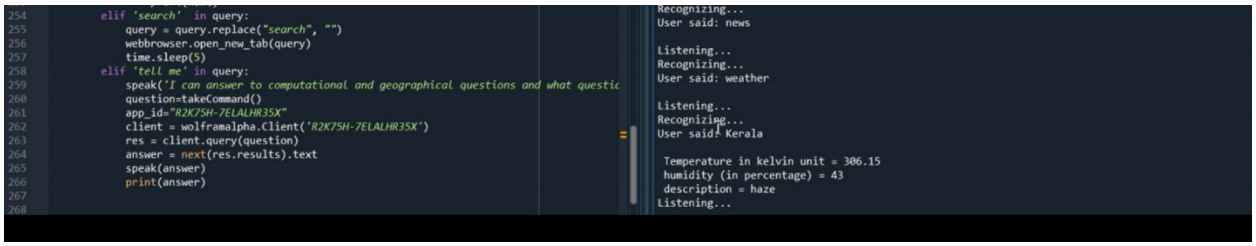
The virtual assistant opens youtube based on command by the user.

d.Get the recent news from ‘Times Of India’



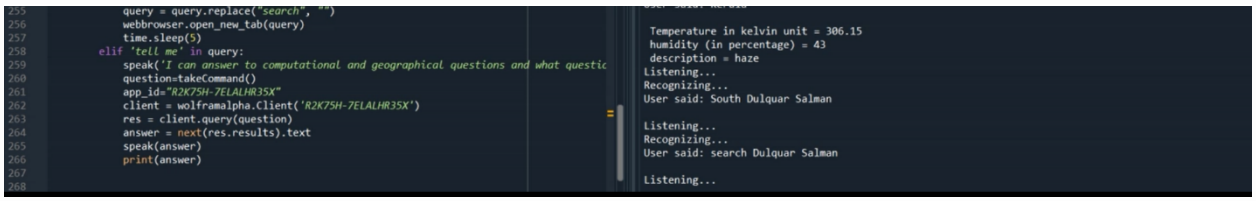
The user commands to get the news and the virtual assistant get the news from 'Times Of India'

e.Get the weather details of a location at a particular instant of time

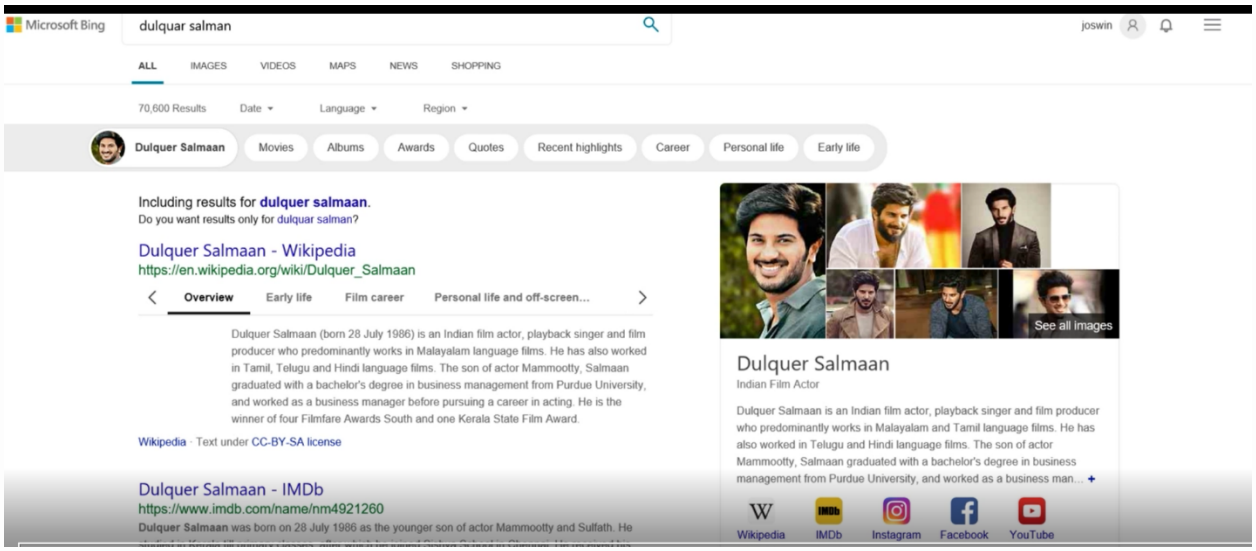


User want to get the weather details of a particular location at an instant of time.Say..kerala…..The virtual assistant responds with the weather details of kerala at that specific instant.

f.Search for a person in google.

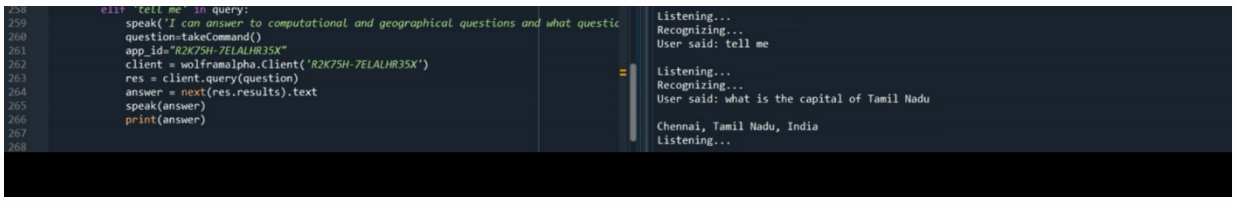


The user wants to search a particular person.



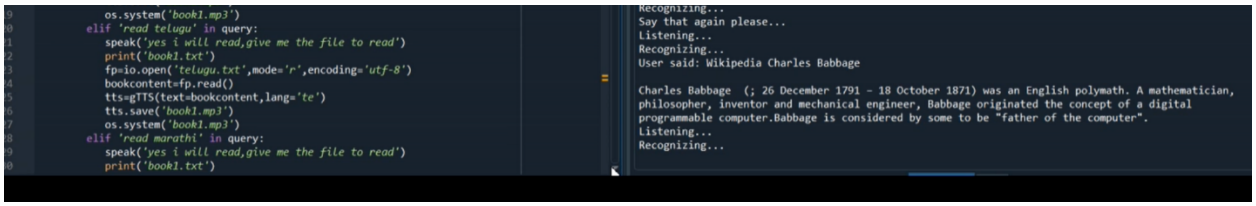
The virtual assistant responds with the results corresponding to that particular person.

g.The user can ask any question and the assistant will answer it.It can answer any geographical and mathematical computaions.



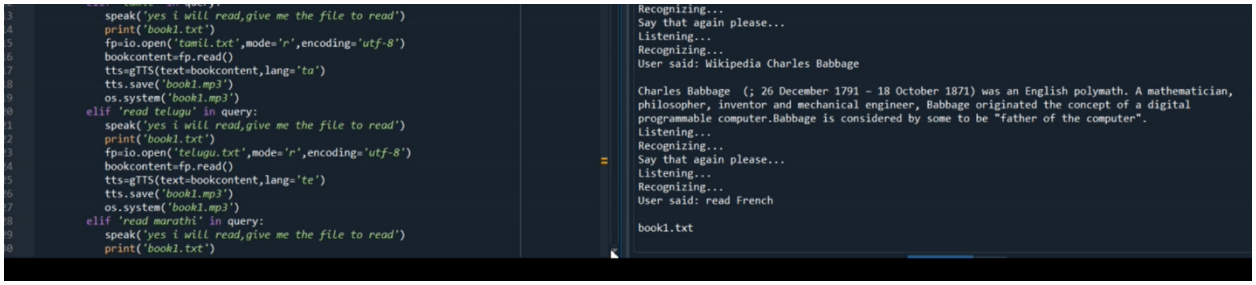
The user asks a particular question and the assistant responds to it.

h.Get details from Wikipedia

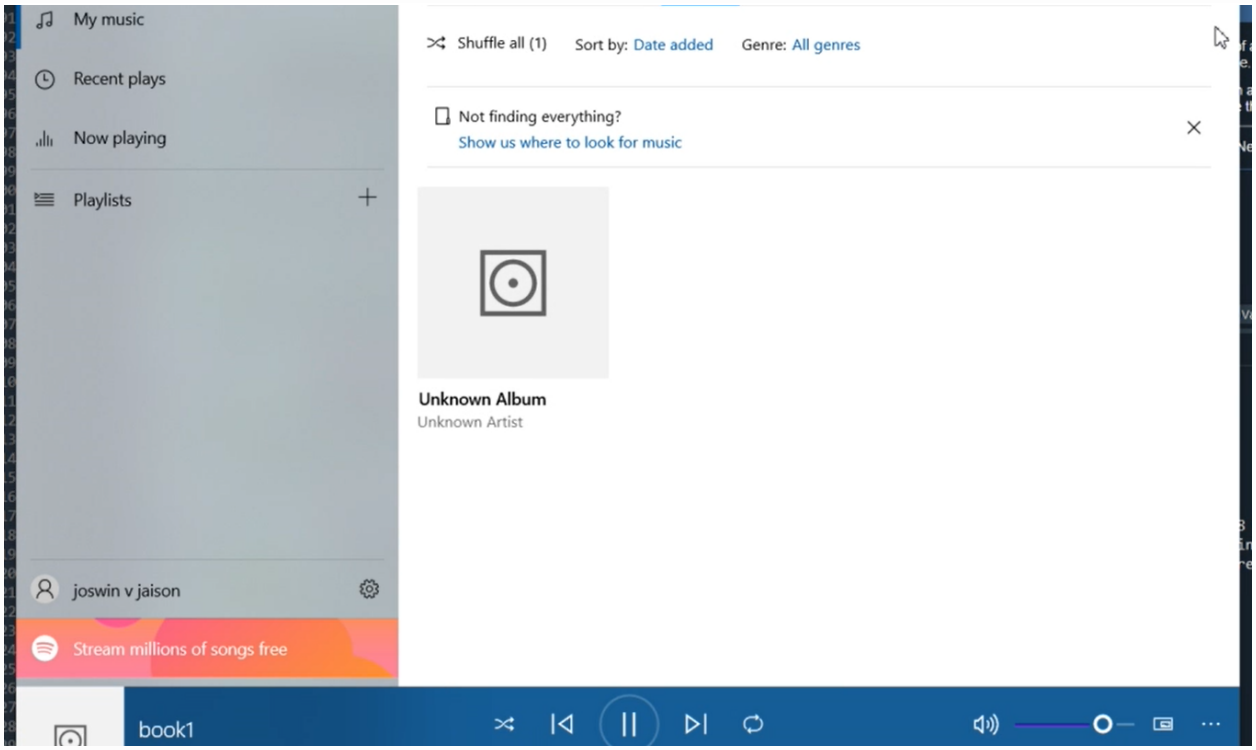


The user wants to get the summary of a person from Wikipedia and the assistant responds to it with the exact answer.

i.Read texts from multiple languages.



The user want the AI assistant to read the text written in French



The program save that as an audio file.By this we can give text written any language to our A.I assistant.

Other functions it will perform:

*Send mail to a particular e-mail id.

*Get the current time .

The following output is verified.

5.CONCLUSION AND FURTHER SCOPE

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.

5.1 FURTHER SCOPE

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.

Functional Requirements:

- Linux Distribution
- Proper Internet Connection
- Github Credentials
- Docker installed
- Python 2.7
- Heroku CLI
- Mplayer for voice support (Text-to-Speech)
- Chromium-based browser, like Chrome, Edge
- Heroku Credentials
- Node JS with npm

The non-functional requirements of the system include:

● The system ensures safety, security and usability, which are

observable during operation (at run time).

● The system is adaptable to different situations.

● The project has good and compact UI using AngularJS with

responsive interface.

● The project is light on resources.

6.REFERENCES

1.How to create your virtual assistant with Python: analytics Vidhya.

2.A project report on Jarvis:AI personal assistant:College of Engineering and Technology ,Bhubaneswar.

3.AI –personal-assistant using Python:mmirthula02

4. Siri, Alexa, and other digital assistants: a study of customer satisfaction with artificial intelligence applications: November 2019Journal of Marketing Management 35(1) DOI: 10.1080/0267257X.2019.1687571 Research Gate.

5. Analyzing the Privacy Attack Landscape for Amazon Alexa Devices: Raphael Leong ryl15@ic.ac.uk Imperial College London

6. 'Alexa, how can we increase trust in you?' An Investigation of Trust in Smart Home Voice Assistants: Author: Rabea Jasmin Adams University of Twente P.O. Box 217, 7500AE Enschede The Netherlands

7. Intelligent Personal Assistant;Academia

8. Survey on Virtual Assistant: Google Assistant, Siri, Cortana, Alexa: 4th International Symposium SIRS 2018, Bangalore, India, September 19–22, 2018, Revised Selected Papers

9. Short Research on Voice Control System Based on Artificial Intelligence Assistant: Publisher: IEEE

10. AI Based Voice Assistant Using Python: Journal of Emerging Technologies and Innovative Research