Voice Recognition based Computer/laptop applications using AI

Submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

by

ANUGULA VENKATESH KUMAR (37130029)
BHAGAVATULA V S S SREEKAR (37130054)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH 2021.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Anugula Venkatesh Kumar (37130029) and Bhagavatula V S S Sreekar (37130054)** who carried out the project entitled "**Voice Recognition based Computer/laptop applications using AI**" under our supervision from **NOVEMBER-2020** to **MARCH-2021.**

Internal Guide
Dr. E. Anna Devi, M.E., Ph.D.

Head of the Department
Dr. T. RAVI M.E, Ph.D.

Submitted for Viva voice examination held on ____________________________

Internal Examiner **External Examiner**

DECLARATION

We **Anugula Venkatesh Kumar (37130029) and Bhagavatula V S S Sreekar (37130054)** hereby declare that the Project Report entitled "**Voice Recognition based Computer/laptop applications using AI**" done by us under the guidance of **Dr. E. Anna Devi** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

1)

2)

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N. NANDITHA, M.E., Ph.D. Dean, School of ELECTRONICS AND COMMUNICATION ENGINEERING** and **Dr. T. RAVI KUMAR, M.E., Ph.D. Head of the Department, Department of ELECTRONICS AND COMMUNICATION ENGINEERING** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to my Project Guide **Dr. E. ANNA DEVI, M.E. Ph.D., Assistant Professor, Department of ELECTRONICS AND COMMUNICATION ENGINEERING** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of the project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **ELECTRONICS AND COMMUNICATION ENGINEERING** who were helpful in many ways for the completion of the project.

We express our gratitude to our parents for their constant encouragement and support for the completion of the project.

Abstract

The project aims to develop a personal-assistant for computer-based system. Jarvis draws its inspiration from virtual assistants like Cortana for Windows, and Siri for iOS. For voice conversion and the voice of the Male / Female Pitch in the pyttsx3 libraries in python and inspired from the Marvel World. Users can interact with the assistant either through voice commands input. As a personal assistant, Jarvis assists the end-user with day-to-day activities like general human conversation, searching queries in google, Bing or yahoo, searching for in YouTube, live weather conditions, opening Microsoft applications like (word, PowerPoint), Playing music and videos, breaking news, calculator, python library install sending mail and reminding the user about the scheduled events and tasks and any other some other features for future purpose. The user statements/ commands are analysed with the help of machine learning to give an optimal solution.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
Introduction

The scenario out of science fiction movies, where we'd come home and just start speaking to our personal home computer or digital assistant which would take care of our every need, is not too far from being realised anymore.Enjoying the luxury of always having an assistant at our side, who can support us in our daily tasks 24/7, is making its way into our lives at a rapid pace of adoption. Granted their functionalities are still limited, but with time voice assistants will be able to help us with more and more activities.The beginnings of the work done in this area reaches back to the 1970. When voice recognition of computers was not much more than a grand vision. It took until Apple's voice assistant Siri entered the stage to gain a significant leap forward because it was the first personal voice assistant that was broadly available to the public as part of the iPhone.

Voice assistants, in general, respond to voice commands and provide the user with specific details about his query. Voice assistants may already process product orders, answer questions, and perform acts such as playing music or initiating a quick phone call with a friend. The technology is already in place, and the next few years will be spent further developing these artificially intelligent assistants, allowing them to perform more complex tasks. Voice assistants' long-term goal is to serve as a smart interface between humans and the immense information and capabilities that the internet provides. Getting rid of the need to communicate with the internet, technology, or other humans in various places using some computer or screen. We'll soon be able to do all with only our words.

Smartspeakers have appeared to have a ton of guarantee in the voice-empowered world. These are outfitted with a receiver for "hearing" and speakers for speaking with us or playing music. Their direct internet link and sophisticated speech recognition tools are the smart parts. Specific commands are understood and responded to by the programme. A good example is asking the smartspeaker's voice assistant about the forecast for tomorrow. The assistant can decode the human voice order and then function on its own by searching the internet for the correct response and reading it back to the user.

The smartspeaker is merely the physical component of the smart assistant, the interface that connects bits and bytes to our human needs and demands. The programme is the brains behind the smartspeaker's artificial intelligence, allowing it to respond to our unique questions and commands. Understanding and processing our requests is the major obstacle that voice assistants must overcome. To have the option to pick the fitting reactions to our orders, a voice collaborator should have the option to learn and utilize dialects.

The smartspeaker's great ability to serve as a bridge between humans and their computers or machines lies in the combination of physical and digital components.

Recent studies show that the rate of adoption of these new smart devices is even faster than it was with the internet or mobile, as humans are getting more used to technology being a consistent part of our lives. Furthermore, talking is just part of human nature and makes it even easier to interact with smartspeakers which are comparatively affordable at around 100 €.

Arising innovations like computer generated simulation, increased reality, and voice collaboration are reshaping individuals' connections with the climate and adjusting intelligent encounters. As a result of progressions in distributed computing, man-made brainpower (AI), and the Internet of Things, voice control is the following stage of human-machine communication (IoT). Because of the broad utilization of cell phones as of late, voice colleagues, for example, Apple's Siri, Google's Assistant, Microsoft's Cortana, and Amazon's Alexa have arisen. Voice colleagues offer administrations to clients utilizing innovation, for example, voice acknowledgment, discourse combination, and Natural Language Processing (NLP). For IoT gadgets that need contact usefulness, a voice interface is required (Metz, 2014). Apart from smartphones, voice assistants are also integrated into smart speakers, which are devices with a microphone and a speaker for communicating with users.

Voice assistants are now available in millions of homes thanks to cloud platforms. Since data must be sent back and forth to centralised data centres, voice assistants rely on a cloud-based architecture. Since a smart speaker is designed to be simple, the majority of the computing and artificial intelligence processing takes place in the cloud rather than on the device itself. The basic concept is that the user makes a request using a voice-activated system, and the voice request is then streamed via the cloud, where it is translated to text. The content solicitation is then shipped off the backend, which measures it and reacts with a book reaction. At long last, the content reaction is shipped off the cloud, where it is changed over into voice and gushed back to the client. The majority of smart speakers lack a screen, but some do, such as the Amazon Echo Show and Echo Spot, the Facebook Portal, and the Google Home Hub.

Since 2017, the success of these devices has been steadily increasing. The introduced base of savvy speakers is projected to hit 225 million by 2020 and 320 million by 2022, as indicated by Canalys (2018). As per juniper research amazon reverberation gadgets and google home gadgets will be the greater part of us family by 2022 and the worldwide will spend on voice collaborators will hit $19 billion by then 2017. With more than 70% of all intelligent voice assistant devices -enabled running on the Alexa platform, it is the industry leader (Griswold, 2018).

1.1 RECENT TRENDS AND FUTURE

7 Key Predictions for Voice In 2020:

Streamlined Conversations

Both Google and Amazon recently announced that both assistants will no longer require the use of repeated “wake” words. Beforehand the two collaborators were subject to a wake word (Alexa or Ok, Google) to start another line of discussion. For instance, one would need to ask "Alexa, what's the current temperature at the corridor indoor regulator?" and afterward need to say, "Alexa" again prior to mentioning that the voice partner to "set the foyer indoor regulator to 23 degrees." It would be more helpful and regular for the client to say, "Alexa, what's the current temperature at the lobby indoor regulator?" and afterward essentially say "set my passage indoor regulator to 23 degrees," without requiring the wake word once more, and well that is conceivable.

Compatibility and Integration

With regards to coordinating voice innovation with different items, Amazon has been on the ball. The individuals who use Alexa will be comfortable with the way that the voice right hand is now incorporated into an immense range of items including Samsung's Family Hub coolers. Google has at last gotten on and has reported Google Assistant Connect. The thought behind this innovation is for producers to make custom gadgets that serve explicit capacities and are coordinated with the Assistant. In 2020, we will see a more noteworthy interest in the advancement of voice-empowered gadgets. This will remember an increment for mid-level gadgets: gadgets that have some associate usefulness however aren't all out shrewd speakers. All things considered, they speak with your brilliant speaker, show or even maybe your telephone over Bluetooth where the preparing occurs on those gadgets. Amazon is now well on its way with an Alexa-empowered divider clock.

Search Behaviors Will Change

Voice quest has become a topic of discussion that has piqued my curiosity. Speech perceptibility would undoubtedly be put to the test. This is due to the absence of a visual interface for speech collaborators. Customers basically can't see or contact a voice interface aside from in the event that it is related with the Alexa or Google Assistant application. Search rehearses, in this way, will see a significant change. Truth be told, if tech research firm Juniper Research is right, voice-based promotion income could reach $19 billion by 2022, thanks in huge part to the development of voice search applications on cell phones.

Individualized Experiences

Voice colleagues will likewise keep on offering more individualized encounters as they improve at separating between voices. Google Home can uphold up to six client accounts and distinguish special voices, which permits Google Home clients to redo numerous highlights. Clients can ask "What's on my schedule today?" or "educate me concerning my day?" and the associate will direct drive times, climate, and news data for singular clients.

Voice Push Notifications

We've recently examined the strategy for utilizing client driven message pop-ups as a way to reconnect clients with your application, voice innovation presents an extraordinary methods for circulating pop-up messages. As an approach to build client commitment and maintenance, pop-up messages essentially help clients to remember the application and show important informing to the client.

Touch Interaction

CES 2019 kept on demonstrating that voice and visual presentations are converging into one consistent experience. This year Google displayed what is being known as the E Ink screen. This presentation can show the climate, nearby traffic data, or schedule occasions. The push to bring visual and voice abilities together permit clients to additionally communicate with the partner.

Security Will Be A Focus

41% of voice right hand clients are worried about trust and security as indicated by a report from Microsoft. With news from Google I/O and Amazon's re:MARS gatherings reporting that colleagues can basically design a whole evening, for instance, discover nearby film times, purchase tickets, book an eatery reservation and timetable a Uber, concerns in regards to installments and delicate data are legitimate. Voice installments, specifically, will turn out to be safer and helpful for clients to make buys. Speaker check and ID will likewise get principal as a component of the voice associate involvement in greater security being worked around the client.

CHAPTER 2

LITERATURE REVIEW

2.1 LITERATURE SURVEY

Very few human assistants have been made using rovers. A reasonable the quantity of works have found in the Literature. Although the techniques and equipment used in this project have already proposed and are in use, we have used unique combinations of algorithms which makes our proposed project novel.

Hill, J., Ford, W.R. and Farreras, I.G. (2015) have done a contrast of human–human online conversations and human–chatbot conversations in real life with artificial intelligence. Human nature and computers. Compared 100 instant message conversations to 100 exchanges with the popular chatbot along seven dimensions. Used to chat in high vocabulary.

G. Bohouta, V. Z. Këpuska 2017 have Comparing Speech Recognition Systems (Microsoft API Google API and CMU Sphinx)", Int. Journal of Engineering Research and Application. With open-source speech recognition systems such as Sphinx-4. The best way to compare automatic speech recognition systems in different environments is by using some audio recordings that were selected from different sources and calculating the word error rate (WER).

Mohasi, L. and Mashao, D. 2006 it is Text-to-Speech Technology in Human-Computer Interaction. In 5th Conference on Human Computer Interaction in Southern Africa. They worked on interacting with robots humans in way like human-to-human communication. They used articulatory speech synthesis, format speech synthesis, conctenative speech synthesis.

Huang, J., Zhou, M. and Yang, D 2007, January. Extracting ChatbotInformation gained from IJCAI's online discussion forums. This paper describes a novel method for collecting high-quality pairs as chat information from online conversation forums in order to facilitate the creation of a chatbot for a specific domain. A cascaded structure is used to retrieve high-quality pairs from a forum. First, an SVM classifier is used to select the responses logically appropriate to the thread title of the root message from all the replies, based on similarities such as structure and text.

Microsoft, Google, apple 2019, Cortona voice assistance, Google assistant, Voice assistance SIRI. These companies came with real-time applications of voice assistance computer based application using API.

2.2 PROBLEM STATEMENTS FROM SURVEY

Many Algorithms and there have been proposals for works. and developed to achieve a better result. Nevertheless, every proposed system lacks a few features. Even though the drawbacks of one system have been compensated in the other, No system as a hole can work autonomously, reducing the effort put in by the user.

Most of the project done in python language. Here we haven't used any API's. The main aim for this project is that all the voice assistants are done using some API. But here the project was handled to be done without any API and using python with some libraries like voice recognition, library for speakers, for text conversion etc..

2.3 OBJECTIVE:

The sole reason of this project is to search for the object specified by the user's voice command, and show the appropriate result to the user. This procedure involves a lot a complication. The audio signals from the voice command has to be converted to machine language to understand the given command. The object from the command has to be understood by the processor and start searching for it.

CHAPTER 3

AIM AND SCOPE

3.1 PROPOSED METHODOLOGIES

- Our proposed project integrates all the requirements to compensate the specified drawbacks
- Geographical questions, weather report, location
- Battery related queries, Opening Microsoft applications
- Change of music track, clear recycle bin
- Create post on facebook
- Opening software applications in Laptop/Computer

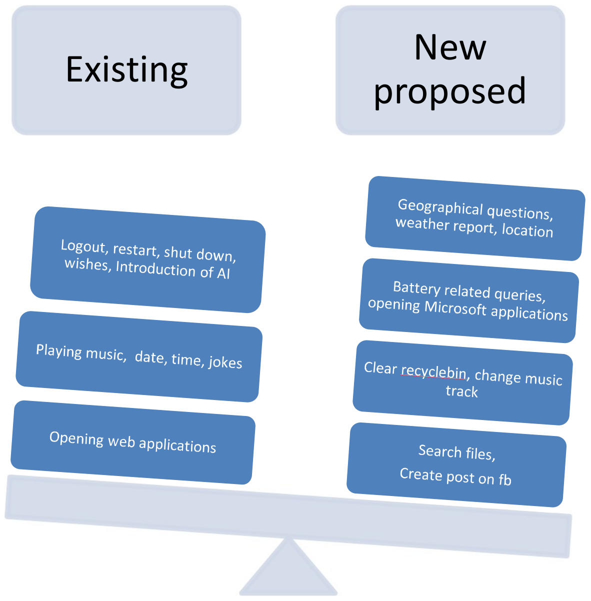

Existing
New proposed
Logout, restart, shut down, wishes, Introduction of AI
Playing music, date, time, jokes
Opening web applications
Geographical questions, weather report, location
Battery related queries, opening Microsoft applications
Clear recyclebin, change music track
Search files, Create post on fb

Fig. 3.1 Existing and proposed methods

3.2 AREA OF PROJECT WORK AND APPLICATION

Voice assistants are programs on digital devices that listen and respond to verbal commands. A client can say,"What's the climate?" and the voice associate will reply with the meteorological forecast for that day and location.They could say, "Reveal to me a story," and the associate will hop into a story. The client could even say, "Request my #1 pizza," and supper will be on its way! Voice aides are so natural to utilize that numerous individuals neglect to pause and WONDER how they work. How voice associates get us? Is it divination? A baffling course of action of codes? A certifiable individual tuning in on the far edge? The fitting reaction is less jumbled than you may might speculate. The app functions similarly to Siri, Google Assistant, and other similar apps. The application's U.I. is self-explanatory and minimal. Voice is recognised as knowledge by it. The framework is being planned so that every one of the administrations given by the cell phones are open by the end client on the client's voice orders. Indeed, I had the comparable idea before I began making my own "Advanced" Personal Assistant. Despite the fact that it isn't just about as proficient and high as like Amazon's Alexa or Google Assistant, Home or Apple's Siri or JARVIS from Iron Man. These days, People are grieved by composing orders into the PC. Be it lingering or a bustling timetable. Composing is a major out of date measure. The answer for this is that we switch over to an associate which gets us and accomplish the underlying work for us. A right hand is the best trade for composing orders. It's named as Desktop Voice Assistant NOVA with Voice Recognition Intelligence, which takes the client contribution to type of client's voice and cycles it and return the yield in different manners like an activity to be performed or the query output is speaked out to the end client.

3.3 SPEECH RECOGNITION TECHNIQUE

Speech recognition, also named as automatic speech recognition (ASR), computer speech recognition, or speech-to-text, is a capability which enables a program to process human speech into a written format. While it’s commonly confused with voice recognition, speech recognition focuses on the translation of speech from a verbal format to a text one whereas voice recognition just seeks to identify an individual user’s voice. Discourse acknowledgment programming (or discourse acknowledgment innovation) empowers telephones, PCs, tablets, and different machines to get, perceive and comprehend human expressions. It utilizes characteristic language as contribution to trigger an activity; empowering our gadgets to likewise react to our verbally expressed orders. The innovation is being utilized to supplant other, more 'drained' techniques for input like composing, messaging, and clicking. A marginally amusing turn of events, seeing as messaging and composing had become the liked technique for correspondence over voice calls only a couple brief years prior.

Today, discourse acknowledgment innovation takes on numerous structures; from directing instant messages to your cell phone while driving, to requesting that your vehicle reserve supper spot at the Chinese eatery as it were, to advising your speaker framework to "if it's not too much trouble, put on that new Beyoncé tune", the capacity to converse with your gadgets have extended to incorporate by far most of innovation that we use in our every day lives. As we remain at the slope of a world destined to be overwhelmed by talking gadgets – and conceivably, advancements with a cognizance – how about we investigate how everything began.

3.4 NATURAL LANGUAGE PROCESSING(NLP)

NLP is a part of man-made brainpower that guides PCs in appreciating, unraveling, and controlling human language. To overcome any issues between human correspondence and PC cognizance, NLP utilizes an assortment of methods, including programming and computational phonetics.

3.4.1 Evolution of Natural Language Processing

Natural language processing isn't a new science, but it's progressing quickly due to a growing interest in human-machine communication, as well as the availability of big data, efficient computing, and improved algorithms.

You can impart in English, Spanish, or Chinese as a person. Be that as it may, the vast majority can't understand a PC's local language, otherwise called machine code or machine language. Not terms, yet a large number of zeros and ones create coherent conduct are utilized to impart at the most minimal levels of your PC.

For sure, software engineers utilized punch cards to speak with the primary PCs 70 years prior. This manual and strenuous interaction was perceived by a moderately modest number of individuals. Presently you can say, "Alexa, I like this melody," and a gadget playing music in your home will bring down the volume and answer, "Alright. Rating saved," in a humanlike voice. At that point it adjusts its calculation to play that tune – and others like it – the following time you tune in to that music station.

We should investigate that connection. Your gadget actuated when it heard you talk, perceived the implicit purpose in the remark, executed an activity and gave criticism in an all around shaped English sentence, all in the space of around five seconds. The total collaboration was made conceivable by NLP, alongside other AI components, for example, AI and profound learning.

3.4.2 How does NLP work?

Dissecting the fundamental components of language

Regular language handling includes a variety of approaches to deciphering human language, ranging from observable and AI methods to rules-based and algorithmic approaches. We need an expansive cluster of approaches on the grounds that the content and voice-based information shifts generally, as do the viable applications. Essential NLP undertakings incorporate tokenization and parsing, lemmatization/stemming, grammatical feature labeling, language discovery and recognizable proof of semantic connections. In the event that you ever charted sentences in grade school, you've done these errands physically previously. In layman's terms, NLP errands break down language into smaller, more natural chunks, seek to understand relations between the chunks, and investigate how the chunks work together to create meaning.

Higher-level NLP skills, such as:

- **Content arrangement**. An etymological based archive synopsis, including search and ordering, content alarms and duplication discovery.

- **Topic revelation and displaying.** Precisely catch the significance and topics in text assortments, and apply progressed investigation to message, similar to advancement and determining.

- **Contextual extraction.** Consequently pull organized data from text-based sources.

- **Sentiment investigation.** Distinguishing the state of mind or emotional conclusions inside a lot of text, including normal notion and assessment mining.

- **Speech-to-text and text-to-discourse transformation.** Changing voice orders into composed content, and the other way around.

- **Document rundown.** Consequently creating rundowns of enormous collections of text.

- **Machine interpretation.** Programmed interpretation of text or discourse starting with one language then onto the next.

The general point in both of these cases is to take crude language information and use etymology and calculations to change or advance it with the goal that it gives really meaning.

3.4.3 Top Benefits of Natural Language Processing (NLP)

The benefits and use of NLP are large and impressive, and only growing by the day. So, what can natural language processing do for your business?

- Perform large-scale analysis
- Get a more objective and accurate analysis
- Streamline processes and reduce costs
- Improve customer satisfaction
- Better understand your market
- Empower your employees
- Gain real, actionable insights

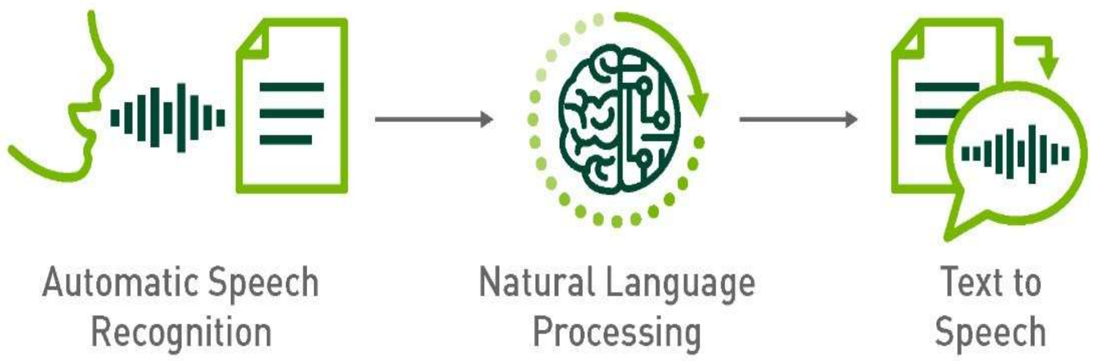


Fig. 3.2 process for NLP

3.5 WORKFLOW

3.5.1 PROPOSED PLAN OF WORK

The work started with analyzing the audio commands given by the user through microphone. This can be in any way similar to getting any data, working PC's inner records, and so forth This is an observational subjective examination, in view of perusing previously mentioned writing and testing their models. Tests are made by programming as indicated by books and online assets, with the express objective to discover best practices and a further developed comprehension of Voice Assistant

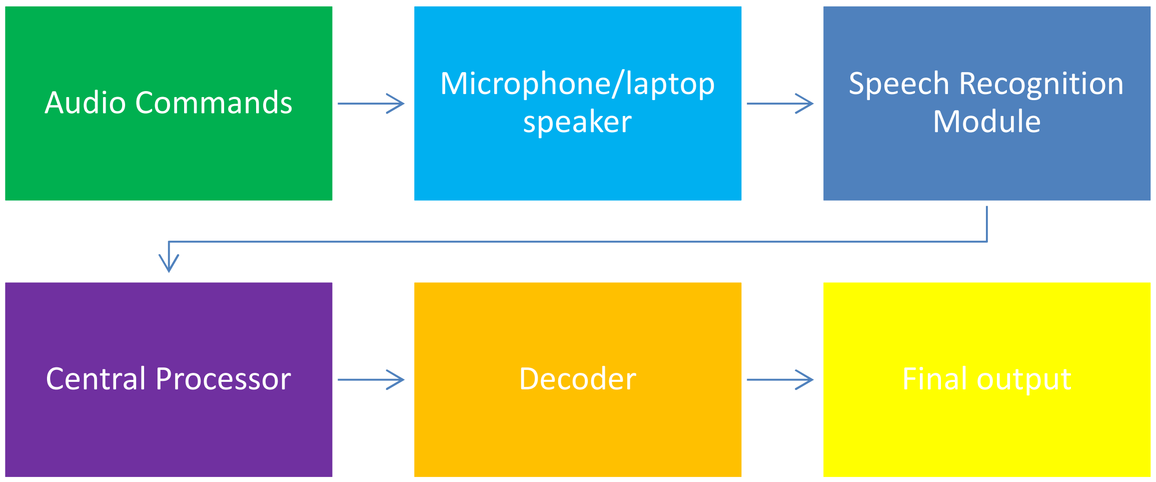


Fig. 3.3 Block diagram for speech recognition

The work process of the fundamental interaction of the voice colleague. Discourse acknowledgment is utilized to change over the discourse contribution to message. This content is then taken care of to the focal processor which decides the idea of the order and calls the significant content for execution. In any case, the intricacies don't stop there. Indeed, even with many long periods of info, different components can assume a gigantic part in whether the product can get you. Foundation commotion can without much of a stretch throw a discourse acknowledgment gadget off course. This is on the grounds that it doesn't naturally can recognize the encompassing sounds it "hears" of a canine yapping or a helicopter flying overhead, from your voice. Designers need to program that capacity into the gadget; they direct information assortment of these encompassing sounds and "tell" the gadget to sift them through. Another factor is the manner in which people normally move the pitch of their voice to oblige for loud conditions; discourse acknowledgment frameworks can be delicate to these pitch changes.

3.6 SOFTWARE AND HARDWARE REQUIREMENTS

3.6.1 Hardware:

LAPTOP/COMPUTER:

- OS – Windows 7, 8 and 10 (32 and 64 bit)
- RAM – 4GB
- PROCESSOR – PENTIUM-4, DUAL CORE, I3, I4,I5,I7,I9

Microphone/speaker

3.6.2 Software:

- PYTHON IDE
- VISUAL STUDIO
- Some Libraries like pyttsx3,Speech Recognition

CHAPTER 4
METHODOLOGY

4.1 PYTHON OVERVIEW

Python is a high-level programming language that is interpreted, interactive, object-oriented, and general-purpose. Guido van Rossum created it between 1985 and 1990. Python source code is also available under the GNU General Public License, much like Perl (GPL).

Python is an undeniable level, deciphered, intelligent and object-arranged scripting language. Python is intended to be exceptionally comprehensible. It utilizes English watchwords much of the time where as different dialects use accentuation, and it has less linguistic developments than different dialects.

• **Python is Interpreted**: Python is prepared at runtime by the translator. You don't have to assemble your program prior to executing it. This is like PERL and PHP.

•**Python is Interactive**: You can really sit at a Python incite and interface with the translator straightforwardly to compose your projects.

•.**Python is a Beginner's Language**: Python is an extraordinary language for the fledgling level software engineers and supports the improvement of a wide scope of utilizations from basic content handling to WWW programs to games.

4.2 PYTHON ENVIRONMENT

Python is accessible on a wide assortment of stages including Linux and Mac OS X. We should see how to set up our Python climate.

4.2.1 History of Python

- Python was created by Guido van Rossum in the last part of the eighties and mid-nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
- Python is gotten from numerous different dialects, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting dialects.
- Python is protected.
- Python is currently kept up by a center advancement group at the foundation, despite the fact that Guido van Rossum actually holds an essential job in coordinating its encouraging.

4.2.2 Characteristics of Python

Following are significant qualities of **Python Programming** −

- It upholds useful and organized programming techniques just as OOP.
- It tends to be utilized as a scripting language or can be gathered to byte-code for building huge applications.
- It gives exceptionally significant level powerful information types and supports dynamic sort checking.
- It upholds programmed trash assortment.

4.2.3 Applications of Python

As referenced previously, Python is perhaps the most generally utilized language over the web. I will list not many of them here:

- **Easy-to-learn** − Python has not many catchphrases, straightforward design, and an unmistakably characterized punctuation. This permits the understudy to get the language rapidly.
- **Easy-to-read** − Python code is all the more plainly characterized and obvious to the eyes.
- **Easy-to-maintain** − Python's source code is genuinely simple to-keep up.
- **A broad standard library** − Python's main part of the library is truly convenient and cross-stage viable on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python's main part of the library is truly convenient and cross-stage viable on UNIX, Windows, and Macintosh.
- **Portable** − Python can run on a wide assortment of equipment stages and has similar interface on all stages.
- **Extendable** − You can add low-level modules to the Python mediator. These modules empower developers to add to or tweak their devices to be more effective.
- **Databases** − Python gives interfaces to all significant business information bases.
- **GUI Programming** − Python upholds GUI applications that can be made and ported to numerous framework calls, libraries and windows frameworks, like Windows MFC, Macintosh, and the X Window arrangement of Unix.
- **Scalable** − Python gives a superior construction and backing for enormous projects than shell scripting.

4.2.4 Decision Making Statements

Dynamic is expectation of conditions happening while execution of the program and indicating activities taken by the conditions.

Choice constructions assess different articulations which produce TRUE or FALSE as result. You need to figure out which move to make and which explanations to execute if result is TRUE or FALSE in any case.

Following is the overall type of a commonplace dynamic design found in the greater part of the programming dialects.

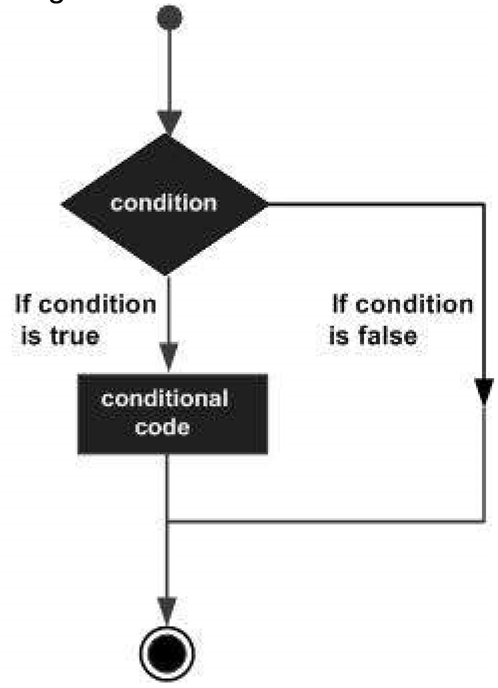


Fig. 4.1 flowchart for if condition

Python programming language accepts any **non-zero** and **non-invalid** qualities as TRUE, and in the event that it is either **zero** or **invalid**, it is expected as FALSE worth.

4.2.5 LOOPS

When all is said in done, articulations are executed successively: The primary assertion in a capacity is executed first, trailed constantly, etc. There might be a circumstance when you need to execute a square of code a few number of times.

Programming dialects give different control structures that consider more convoluted execution ways.

A circle articulation permits us to execute an assertion or gathering of proclamations on various occasions. The accompanying graph outlines a circle proclamation.

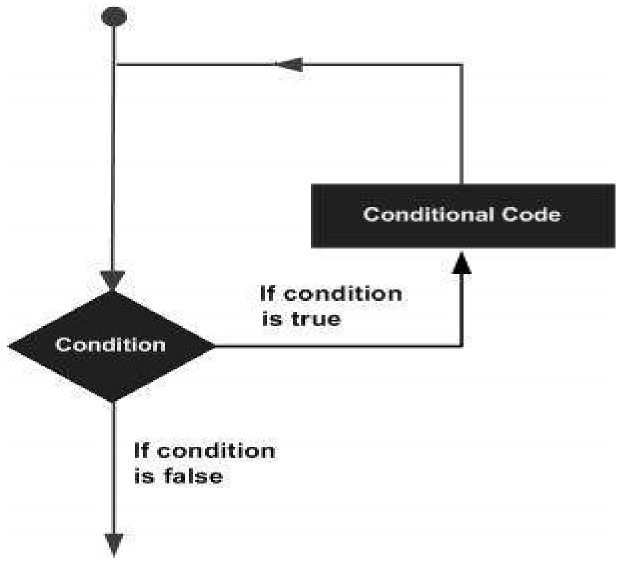


Fig. 4.2 flowchat for working of if loop

4.2.6 FUNCTIONS

A capacity is a square of coordinated, reusable code that is utilized to play out a solitary, related activity. Capacities give better particularity to your application and a serious level of code reusing.

As you definitely know, Python gives you many underlying capacities like print(), and so forth yet you can likewise make your own capacities. These capacities are called client characterized capacities.

Defining a Function

You can characterize capacities to give the necessary usefulness. Here are basic guidelines to characterize a capacity in Python.

- Function blocks start with the watchword def followed by the capacity name and brackets (( )).
- Any input boundaries or contentions ought to be set inside these brackets. You can likewise characterize boundaries inside these enclosures.
- The first explanation of a capacity can be a discretionary assertion - the documentation line of the capacity or docstring.
- The code block inside each capacity begins with a colon (:) and is indented.

- The articulation return [expression] exits a capacity, alternatively passing back an articulation to the guest. A return explanation without any contentions is equivalent to bring None back.

Calling a Function

- Defining a capacity just gives it a name, indicates the boundaries that are to be remembered for the capacity and designs the squares of code.
- Once the fundamental construction of a capacity is finished, you can execute it by calling it from another capacity or straightforwardly from the Python brief. Following is the guide to call printme()

4.2.7MODULES

- A module permits you to sensibly coordinate your Python code. Gathering related code into a module makes the code more obvious and use. A module is a Python object with discretionarily named ascribes that you can tie and reference.
- Simply, a module is a record comprising of Python code. A module can characterize capacities, classes and factors. A module can likewise incorporate runnable code.

4.3 LIBRARIES

There are a few sorts of libraries are accessible in python. They are

4.3.1. Python's standard library

• Pandas
• NumPy
• Sklearn
• seaborn
• matplotlib
• Importing Datasets
• OpenCV

NUMPY

NumPy is one such amazing library for cluster handling alongside a huge assortment of significant level numerical capacities to work on these exhibits. These capacities fall into classes like Linear Algebra, Trigonometry, Statistics, Matrix control, and so on

Getting NumPy
NumPy's primary item is a homogeneous multidimensional exhibit. Dissimilar to python's cluster class which just handles one-dimensional exhibit, NumPy's Nd cluster class can deal with multidimensional cluster and gives greater usefulness. NumPy's measurements are known as tomahawks.

For instance, the exhibit beneath has 2 measurements or 2 tomahawks in particular lines and segments. Some of the time measurement is otherwise called a position of that specific exhibit or network.

Importing NumPy
NumPy is imported utilizing the accompanying order. Note here np is the show followed for the moniker so we don't have to compose NumPy without fail.

• import numpy as np

NumPy is the fundamental library for logical calculations in Python and this article delineates a portion of its most as often as possible utilized capacities. Understanding NumPy is the principal significant advance in the excursion of AI and profound learning.

Pandas:
Pandas is an open source Python bundle that is most generally utilized for information science/information examination and AI assignments. It is based on top of another bundle named Numpy, which offers help for multi-dimensional clusters. As perhaps the most famous information fighting bundles, Pandas functions admirably with numerous other information science modules inside the Python environment, and is ordinarily remembered for each Python conveyance, from those that accompany your working framework to business merchant disseminations like ActiveState's Active Python**.**
Loading and Saving Data with Pandas
At the point when you need to utilize Pandas for information investigation, you'll as a rule use it in one of three unique ways:

- Convert a Python's rundown, word reference or Numpy cluster to a Pandas information outline.
- Open a nearby record utilizing Pandas, generally a CSV document, however could likewise be a delimited book document (like TSV), Excel, and so on.
- Open a distant document or information base like a CSV or a JSONon a site through a URL or read from a SQL table/data set.

There are various orders to every one of these choices, however when you open a document, they would resemble this:

pd.read_filetype()

Viewing and Inspecting Data

Since you've stacked your information, it's an ideal opportunity to investigate. How does the information outline look? Running the name of the information casing would give you the whole table, yet you can likewise get the main n lines with df.head(n) or the keep going n columns with df.tail(n). df.shape would give you the quantity of lines and sections. df.info() would give you the record, datatype and memory data. The order s.value_counts(dropna=False) would permit you to see remarkable qualities and means an arrangement (like a segment or a couple of sections). An extremely helpful order is df.describe() which inputs synopsis measurements for mathematical sections. It is additionally conceivable to get insights on the whole information outline or an arrangement (a section and so forth):

- df.mean()Returns the mean, all things considered.
- df.corr()Returns the relationship between's sections in an information outline
- df.count()Returns the quantity of non-invalid qualities in every information outline segment
- df.max()Returns the most noteworthy worth in every section
- df.min()Returns the lowest noteworthy in every column
- df.median()Returns the middle of every section
- df.std()Returns the standard deviation of every section

SKLEARN:

Scikit-learn (Sklearn) is the most valuable and strong library for AI in Python. It gives a choice of effective apparatuses for AI and factual demonstrating including order, relapse, bunching and dimensionality decrease through a consistence interface in Python. This library, which is generally written in Python, is based upon NumPy, SciPy and Matplotlib.

Prerequisites

- Before we begin utilizing scikit-learn most recent delivery, we require the accompanying –
- Python (>=3.5)
- NumPy (>= 1.11.0)
- Scipy (>= 0.17.0)li
- Joblib (>= 0.11)
- Matplotlib (>= 1.5.1) is required for Sklearn plotting capabilities.
- Pandas (>= 0.18.0) is required for some of the scikit-learn examples using data structure and analysis.

Installation

On the off chance that you previously introduced NumPy and Scipy, following are the two most effortless approaches to introduce scikit-learn −**Using pip**

Following order can be utilized to introduce scikit-learn by means of pip –
pip install -U scikit-learn

Features

Maybe than zeroing in on stacking, controlling and summing up information, Scikit-learn library is centered around displaying the information. Probably the most mainstream gatherings of models given by Sklearn are as per the following −

Supervised Learning algorithms − Practically every one of the well known directed learning calculations, as Linear Regression, Support Vector Machine (SVM), Decision Tree and so on, are the piece of scikit-learn.

Unsupervised Learning algorithms − Then again, it likewise has all the well known unaided taking in calculations from bunching, factor investigation, PCA (Principal Component Analysis) to solo neural organizations.

Clustering − This model is utilized for gathering unlabeled information.

Cross Validation − It is utilized to check the precision of directed models on inconspicuous information.

Dimensionality Reduction − It is utilized for decreasing the quantity of characteristics in information which can be additionally utilized for summarisation, representation and highlight determination.

Ensemble methods − As name recommend, it is utilized for consolidating the expectations of various administered models.

Feature extraction − It is utilized to separate the highlights from information to characterize the ascribes in picture and text information.

Feature selection − It is utilized to recognize helpful ascribes to make managed models.

Open Source − It is open source library and furthermore financially usable under BSD permit.

MATPLOTLIB:

Matplotlib is a stunning perception library in Python for 2D plots of clusters. Matplotlib is a multi-stage information representation library based on NumPy exhibits and intended to work with the more extensive SciPy stack. It was presented by John Hunter in the year 2002.

Perhaps the best advantage of perception is that it permits us visual admittance to gigantic measures of information in effectively absorbable visuals. Matplotlib comprises of a few plots like line, bar, dissipate, histogram and so forth.

Installation:

Windows, Linux and macOS circulations have matplotlib and the vast majority of its conditions as wheel bundles. Run the accompanying order to introduce matplotlib bundle:

python -mpip install -U matplotlib

Importing matplotlib :

from matplotlib import pyplot as plt

or

import matplotlib.pyplot as plt

Basic plots in Matplotlib :
Matplotlib accompanies a wide assortment of plots. Plots assists with getting patterns, designs, and to make relationships. They're normally instruments for thinking about quantitative data. A portion of the example plots are covered here.

4.3.2. EXTERNAL LIBRARIES ARE USED USED:

- subprocess
- pyttsx3
- datetime
- time
- speech_recognition
- wikipedia
- smtplib
- webbrowser
- wikipedia
- pyautogui
- psutil
- pyjokes
- random
- requests
- gTTS
- playsound
- Selenium
- pygame
- pickle

These are the libraries used in our code. Each library has some specified function.

SUBPROCESS:

The subprocess module permits you to generate new cycles, associate with their info/yield/mistake pipes, and get their bring codes back. This module means to supplant a few more seasoned modules and capacities:

- os.system
- os.spawn*

Data about how the subprocess module can be utilized to supplant these modules and capacities can be found in the accompanying areas.

Using the subprocess Module

- The prescribed way to deal with conjuring subprocesses is to utilize the run() work for all utilization cases it can deal with. For further developed use cases, the basic Popen interface can be utilized straightforwardly.
- The run() work was included Python 3.5; on the off chance that you need to hold similarity with more seasoned adaptations, see the Older undeniable level API area.
- subprocess.run(args, *, stdin=None, input=None, stdout=None, stderr=None, capture_output=False, shell=False, cwd=None, timeout=None, check=False, encoding=None, errors=None, text=None, env=None, universal_newlines=None, **other_popen_kwargs)
- Run the order depicted by args. Trust that order will finish, at that point return a CompletedProcess occasion.
- The contentions appeared above are simply the most well-known ones, portrayed underneath in Frequently Used Arguments (consequently the utilization of watchword just documentation in the condensed signature). The full capacity mark is to a great extent equivalent to that of the Popen constructor - the greater part of the contentions to this capacity are gone through to that interface. (break, info, check, and capture_output are most certainly not.)
- If capture_output is valid, stdout and stderr will be caught. At the point when utilized, the inward Popen object is naturally made with stdout=PIPE and stderr=PIPE. The stdout and stderr contentions may not be provided simultaneously as capture_output. On the off chance that you wish to catch and join the two streams into one, use stdout=PIPE and stderr=STDOUT rather than capture_output.
- The break contention is passed to Popen.communicate(). In the event that the break terminates, the youngster interaction will be murdered and hung tight for. The TimeoutExpired exemption will be re-brought up after the youngster cycle has ended.
- The input contention is passed to Popen.communicate() and along these lines to the subprocess' stdin. Whenever utilized it should be a byte succession, or a string if encoding or blunders is determined or text is valid. At the point when utilized, the interior Popen object is naturally made with stdin=PIPE, and the stdin contention may not be utilized also.
- If check is valid, and the interaction exits with a non-zero leave code, a CalledProcessError special case will be raised. Characteristics of that exemption hold the contentions, the leave code, and stdout and stderr on the off chance that they were caught.
- If encoding or blunders are determined, or text is valid, record objects for stdin, stdout and stderr are opened in text mode utilizing the predefined encoding and mistakes or the io.TextIOWrapper default.

  The universal_newlines contention is identical to message and is accommodated in reverse similarity. Naturally, record objects are opened in paired mode.

- If env isn't None, it should be a planning that characterizes the climate factors for the new cycle; these are utilized rather than the default conduct of acquiring the current interaction's current circumstance. It is passed straightforwardly to Popen.

Gtts

- The content to-discourse (TTS) is the way toward changing over words into a vocal sound structure. The program, apparatus, or programming takes an information text from the client, and utilizing techniques for common language handling comprehends the phonetics of the language being utilized, and performs intelligent induction on the content. This prepared content is passed into the following square where advanced sign handling is performed on the handled content. Utilizing numerous calculations and changes this prepared content is at long last changed over into a discourse design. This whole cycle includes the orchestrating of discourse. The following is a basic square outline to comprehend something similar.
- Text-to-discourse (TTS) innovation peruses resoundingly computerized text. It can take words on PCs, cell phones, tablets and convert them into sound. Likewise, a wide range of text records can be perused resoundingly, including Word, pages archive, online site pages can be perused so anyone might hear. TTS can help kids who battle with perusing. Numerous apparatuses and applications are accessible to change over text into discourse.

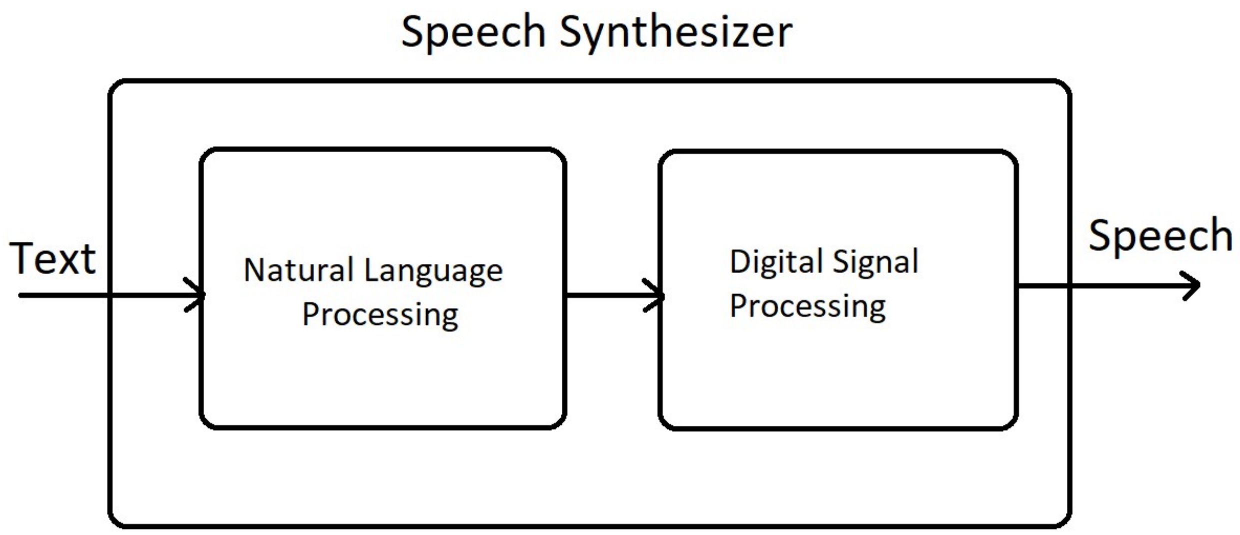


Fig. 4.3 working of speech synthesizer

This seems like a serious convoluted interaction, yet on account of python and the gTTS module, this cycle can be rearranged to only a couple lines of code. From the square outline, we can comprehend that the content being passed is right off the bat pre-handled with the assistance of regular language preparing, and afterward utilizing computerized signal handling is changed over to discourse.

Right away, how about we get our hands filthy with some code

Installation of gTTS module:

We will be working with python for the remainder of the instructional exercise. On the off chance that you haven't introduced python as of now, kindly do as such here. The establishment of the gTTS module is straightforward and should be possible utilizing the accompanying order in the order brief terminal —

```
pip introduce gTTS
```

After the establishment is done, we can continue to compose a basic program to see how precisely we can utilize this module to change over our composed content into a discourse changed over yield. Open the python document and give it a name of your decision and ensure it closes with the .py design.

```
from gtts import gTTS

text = "Hello! My name is Bharath."
tts = gTTS(text)
tts.save("hi.mp3")
```

gTTS (Google Text-to-Speech)is a Python library and CLI apparatus to interface with Google Translate text-to-discourse API. We will import the gTTS library from the gtts module which can be utilized for discourse translation.The text variable is a string used to store the client's information. The content can be supplanted by anything of your decision inside the statements. Another option can be to utilize the information explanation for the client to type their own ideal information each time the program is run. This should be possible as follows:

```
text = input("Enter your text: ")
tts = gTTS(text)
tts.save("user_input.mp3")
```

Conclusion:

The gTTS module can be utilized broadly on different dialects like French, German, Hindi, and so on, also. This is incredibly valuable when there is a correspondence boundary and the client can't pass on his messages to individuals. Text-to-discourse is an incredible assistance to the outwardly hindered individuals or individuals with different inabilities as it can help them by aiding the content to discourse interpretation. There are likewise numerous thoughts conceivable with the gTTS module and it tends to be utilized for different dialects also.

There is potential for a ton of marvelous tasks with something similar. I will urge watchers to take a stab at testing around additional with this module. Watchers can don't hesitate to allude here for a cool task which I had recently done. In this arrangement, we will perceive how we can carry out an interpretation utilizing profound learning and how the gTTS module will assume a part in doing this.

PTTYSX3:

Using pyttsx3

The gTTS module can be utilized broadly on different dialects like French, German, Hindi, and so on, also. This is incredibly valuable when there is a correspondence boundary and the client can't pass on his messages to individuals. Text-to-discourse is an incredible assistance to the outwardly hindered individuals or individuals with different inabilities as it can help them by aiding the content to discourse interpretation. There are likewise numerous thoughts conceivable with the gTTS module and it tends to be utilized for different dialects also.

There is potential for a ton of marvelous tasks with something similar. I will urge watchers to take a stab at testing around additional with this module. Watchers can don't hesitate to allude here for a cool task which I had recently done. In this arrangement, we will perceive how we can carry out an interpretation utilizing profound learning and how the gTTS module will assume a part in doing this.

The Engine factory

pyttsx3.**init**([*driverName : string*, *debug : bool*]) → pyttsx3.Engine

> Gets a reference to a motor case that will utilize the given driver. On the off chance that the mentioned driver is as of now being used by another motor case, that motor is returned. Something else, another motor is made.

Table 4.1 parameters and raises for pttysx3

Parameters:	**driverName** – Name of the pyttsx3.drivers module to load and utilize. Defaults to the best accessible driver for the stage, at present: *sapi5* - SAPI5 on Windows *nsss* - NSSpeechSynthesizer on Mac OS X *espeak* - eSpeak on every other platform **debug** – Enable debug output or not.
Raises:	**ImportError** – When the requested driver is not found **RuntimeError** – When the driver fails to initialize

The Engine interface

class pyttsx3.engine.**Engine**

Gives application admittance to message to-discourse combination.

connect(topic : string, cb : callable) → dict

Registers a callback for warnings on the given theme.

Table 4.2 for pttysx3 engine

Parameters:	**topic** – Name of the event to subscribe to. **cb** – Function to invoke when the event fires.
Returns:	A token that the caller can use to unsubscribe the callback later.

Coming up next are the legitimate points and their callback marks.

started-utterance

Terminated when the motor starts talking an expression. The related callback should have the folowing mark.

onStartUtterance(*name : string*) → None

- **Parameters:** **name** – Name associated with the utterance.

started-word

Terminated when the motor starts talking a word. The related callback should have the folowing mark.

onStartWord(*name : string*, *location : integer*, *length : integer*)

- **Parameters:** **name** – Name associated with the utterance.

finished-utterance

Terminated when the motor wraps up talking an expression. The related callback should have the folowing mark.

onFinishUtterance(*name : string*, *completed : bool*) → None

- **Parameters:** **name** – Name associated with the utterance.
  completed – True if the utterance was output in its

entirety or not.

error
Fired when the engine encounters an error. The associated callback must have the folowing signature.

onError(*name : string*, *exception : Exception*) → None

- **Parameters:** **name** – Name associated with the utterance that caused the error.
  exception – Exception that was raised.

disconnect(*token : dict*)
Unregisters a notification callback.

- **Parameters:** **token** – Token returned by connect() associated with the callback to be disconnected.

endLoop() → None

Closures a running occasion circle. On the off chance that startLoop() was called with useDriverLoop set to True, this strategy quits preparing of motor orders and promptly leaves the occasion circle. On the off chance that it was called with False, this strategy quits preparing of motor orders, however it is dependent upon the guest to end the outer occasion circle it began.

- **Raises:** **RuntimeError** – When the loop is not running

getProperty(*name : string*) → object

PGets the current value of an engine property.

- **Parameters:** **name** – Name of the property to query.
- **Returns:** Value of the property at the time of this invocation.

The accompanying property names are substantial for all drivers.

rate

The accompanying property names are substantial for all drivers.

voice

String identifier of the dynamic voice.

voices

Rundown of pyttsx3.voice.Voice descriptor objects.

volume

Coasting direct volume in the scope of 0.0 toward 1.0 comprehensive. Defaults to 1.0..

isBusy() → bool

Gets if the engine is currently busy speaking an utterance or not.

- **Returns:** True if speaking, false if not.

runAndWait() → None

Squares while handling all as of now lined orders. Summons callbacks for motor warnings fittingly. Returns when all orders lined before this call are exhausted from the line.

say(*text : unicode*, *name : string*) → None

Squares while taking care of all as of now lined requests. Request callbacks for engine admonitions fittingly. Returns when all orders lined before this call are depleted from the line.

- **Parameters:** **text** – Text to speak.
  name – Name to associate with the utterance. Included in notifications about this utterance.

setProperty(*name*, *value*) → None

Lines an order to set a motor property. The new property estimation influences all expressions lined after this order.

- **Parameters:** **name** – Name of the property to change.
  value – Value to set.

The following property names are valid for all drivers.

rate

Integer speech rate in words per minute.

voice

String identifier of the active voice.

volume

Floating point volume in the range of 0.0 to 1.0 inclusive.

startLoop([*useDriverLoop : bool*]) → None

Starts running an occasion circle during which lined orders are handled and notices are terminated.

• **Parameters:**	**useDriverLoop** –Consistent with utilize the circle given by the chose driver. Bogus to show the guest will enter its own circle in the wake of conjuring this technique. The guest's circle should siphon occasions for the driver being used so that pyttsx3 warnings are conveyed appropriately (e.g., SAPI5 requires a COM message siphon). Defaults to True.

stop() → None

Stops the current expression and clears the order line.

The Voice metadata

class pyttsx3.voice.

Voice

Contains data about a discourse synthesizer voice.

age

Whole number age of the voice in years. Defaults to None if obscure.

sex

String sex of the voice: male, female, or fair-minded. Defaults to None if dark.

Id

String identifier of the voice. Used to set the dynamic voice by means of pyttsx3.engine.Engine.setPropertyValue(). This characteristic is constantly characterized.

Languages

Rundown of string dialects upheld by this voice. Defaults to an unfilled rundown of obscure.

Name

Comprehensible name of the voice. Defaults to None if obscure.

Example

Speaking text

```
import pyttsx3

engine = pyttsx3.init()

engine.say(‘Sally sells seashells by the seashore.’)

engine.say(‘The quick brown fox jumped over the lazy dog.’)

engine.runAndWait()
```

Saving voice to a file

```
import pyttsx3

engine = pyttsx3.init()

engine.save_to_file(‘Hello World’ , ‘test.mp3’)

engine.runAndWait()
```

Listening for events

```
import pyttsx3
```

```
def onStart(name):
   print ‘starting’, name
def onWord(name, location, length):
   print ‘word’, name, location, length
def onEnd(name, completed):
   print ‘finishing’, name, completed
engine = pyttsx3.init()
engine.connect(‘started-utterance’, onStart)
engine.connect(‘started-word’, onWord)
engine.connect(‘finished-utterance’, onEnd)
engine.say(‘The quick brown fox jumped over the lazy dog.’)
engine.runAndWait()
```

Interrupting an utterance

```
import pyttsx3
def onWord(name, location, length):
   print ‘word’, name, location, length
   if location > 10:
      engine.stop()
engine = pyttsx3.init()
engine.connect(‘started-word’, onWord)
engine.say(‘The quick brown fox jumped over the lazy dog.’)
engine.runAndWait()
```

Changing voices

```
engine = pyttsx3.init()
voices = engine.getProperty(‘voices’)
```

```
for voice in voices:
   engine.setProperty('voice', voice.id)
   engine.say('The quick brown fox jumped over the lazy dog.')
engine.runAndWait()
```

Changing speech rate

```
engine = pyttsx3.init()
rate = engine.getProperty('rate')
engine.setProperty('rate', rate+50)
engine.say('The quick brown fox jumped over the lazy dog.')
engine.runAndWait()
```

Changing volume

```
engine = pyttsx3.init()
volume = engine.getProperty('volume')
engine.setProperty('volume', volume-0.25)
engine.say('The quick brown fox jumped over the lazy dog.')
engine.runAndWait()
```

Running a driver event loop

```
engine = pyttsx3.init()
def onStart(name):
   print 'starting', name
def onWord(name, location, length):
   print 'word', name, location, length
def onEnd(name, completed):
   print 'finishing', name, completed
```

```
  if name == ‘fox’:
    engine.say(‘What a lazy dog!’, ‘dog’)
  elif name == ‘dog’:
    engine.endLoop()
```

Using an external event loop

```
engine = pyttsx3.init()
engine.say(‘The quick brown fox jumped over the lazy dog.’, ‘fox’)
engine.startLoop(False)
# engine.iterate() must be called inside externalLoop()
externalLoop()
engine.endLoop()
```

Speech Recognition

Discourse acknowledgment, as the name proposes, alludes to programmed acknowledgment of human discourse. Discourse acknowledgment is perhaps the main errands in the space of human PC connection. On the off chance that you have at any point interfaced with Alexa or have at any point requested Siri to finish an undertaking, you have effectively encountered the force of discourse acknowledgment. Discourse acknowledgment has different applications going from programmed record of discourse information (like voice messages) to collaborating with robots by means of discourse.

In this instructional exercise, you will perceive how we can build up an exceptionally basic discourse acknowledgment application that is fit for perceiving discourse from sound records, just as live from a receiver. Along these lines, we should start right away.

A few discourse acknowledgment libraries have been created in Python. Anyway we will utilize the SpeechRecognition library, which is the easiest of the multitude of libraries.

Installing SpeechRecognition Library

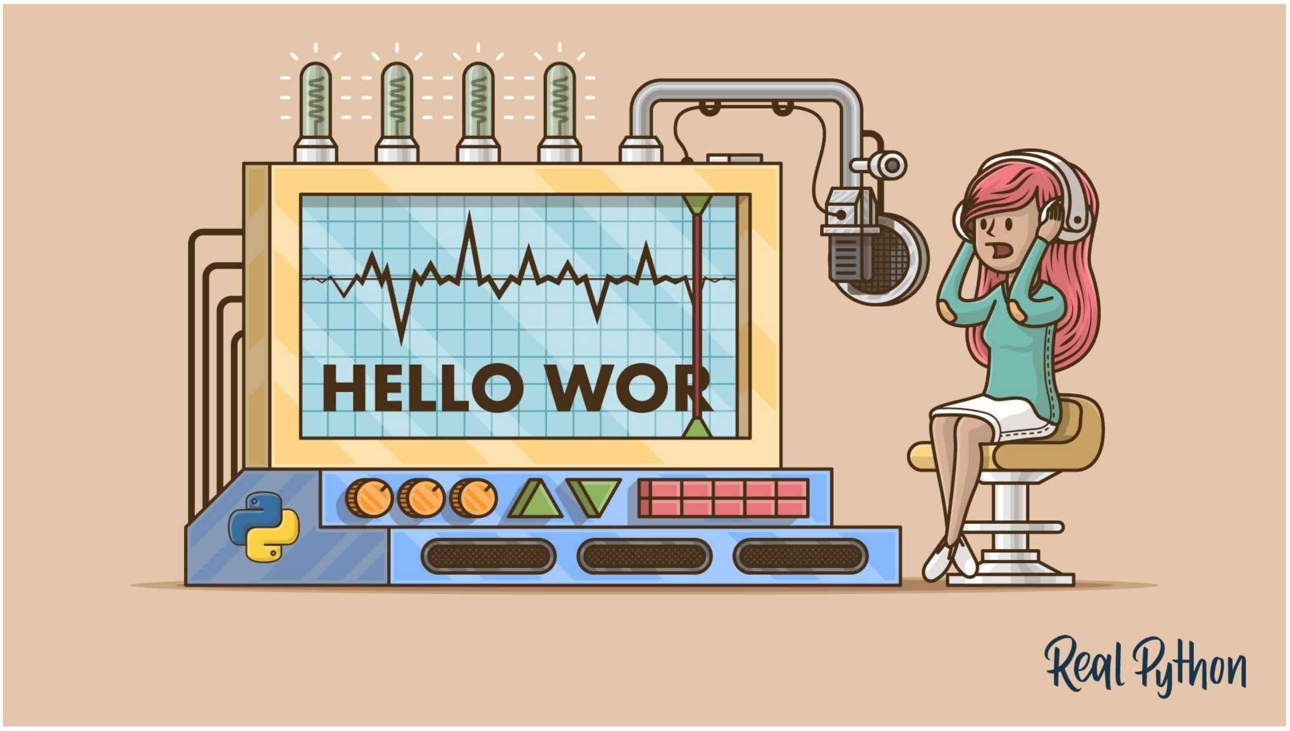


Fig. 4.4 speech recognition

pip install SpeechRecognition

Speech Recognition from Audio Files

In this segment, you will perceive how we can interpret discourse from a sound document to message. The sound record that we will use as information can be downloaded from this connection. Download the record to your neighborhood document framework. The initial step, as usual, is to import the necessary libraries. For this situation, we just need to import the discourse acknowledgment library that we just downloaded.

import speech_recognition as speech_recog

To change discourse over to message the unrivaled class we need is the Recognizer class from the speech_recognition module. Contingent on the hidden API used to change discourse over to message, the Recognizer class has following strategies:

- recognize_bing(): Uses Microsoft Bing Speech API
- recognize_google(): Uses Google Speech API
- recognize_google_cloud(): Uses Google Cloud Speech API
- recognize_houndify(): Uses Houndify API by SoundHound
- recognize_ibm(): Uses IBM Speech to Text API
- recognize_sphinx(): Uses PocketSphinx API

Among the entirety of the above strategies, the recognize_sphinx() technique can be utilized disconnected to make an interpretation of discourse to message.

To perceive discourse from a sound document, we need to make an object of the AudioFile class of the speech_recognition module. The way of the sound record that you need to mean content is passed to the constructor of the AudioFile class. Execute the accompanying content:

```
sample_audio = speech_recog.AudioFile('E:/Datasets/my_audio.wav')
```

In the above code, update the way to the sound document that you need to translate.

We will utilize the recognize_google() technique to decipher our sound records. Nonetheless, the recognize_google() strategy requires the AudioData object of the speech_recognition module as a boundary. To change over our sound document to an AudioData object, we can utilize the record() technique for the Recognizer class. We need to pass the AudioFile object to the record() technique, as demonstrated beneath

:

```
with sample_audio as audio_file:
    audio_content = recog.record(audio_file)
```

Presently in the event that you check the kind of the audio_content variable, you will see that it has the sort speech_recognition.AudioData.

```
type(audio_content)
```

Output:

```
speech_recognition.AudioData
```

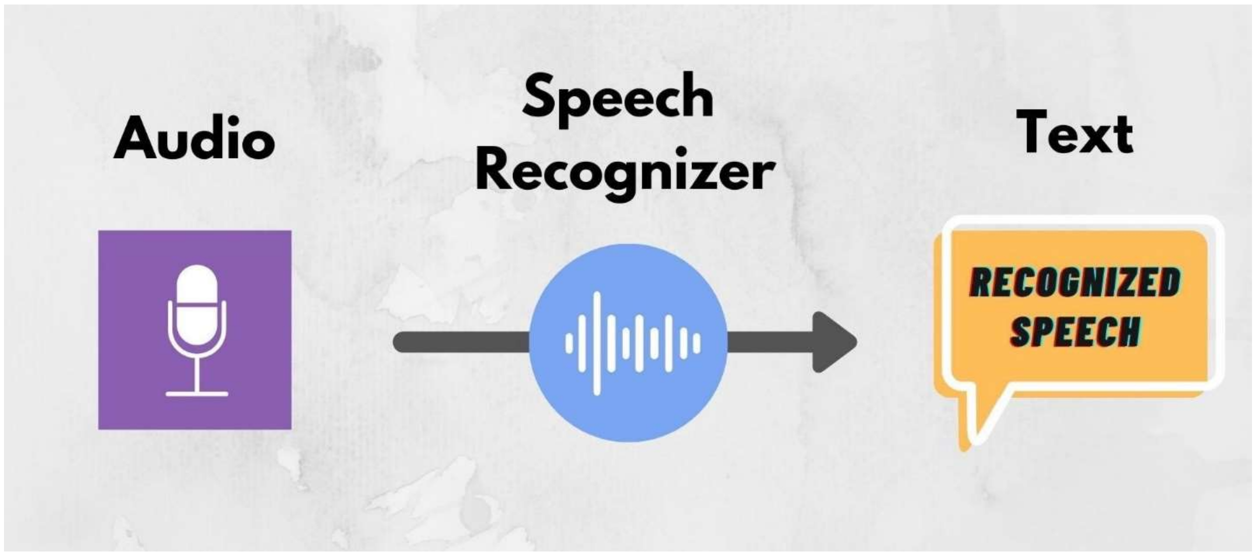


Fig. 4.5 audio to speech conversion

JSON:

JSON is a standard organization for information trade, which is enlivened by JavaScript. By and large, JSON is in string or text design. JSON represents JavaScript Object Notation.

The linguistic structure of JSON: JSON is composed as key and worth pair.

```
{
        "Key":  "Value",
        "Key":  "Value",
}
```

JSON is very much like Python word reference. Python upholds JSON, and it has an inbuilt library as a JSON.

JSON Library in Python

'marshal' and **'pickle'** outer modules of Python keep a variant of JSON library. To perform JSON related tasks like encoding and interpreting in Python you need first to import JSON library and for that in your .py record, import json

Table 4.3 Json Library

Method	**Description**
dumps()	encoding to JSON objects
dump()	encoded string writing on file
loads()	Decode the JSON string
load()	Decode while JSON file read

Python to JSON (Encoding)

JSON Library of Python performs following interpretation of Python objects into JSON objects as a matter of course

Table 4.4 Json objects(encoding)

Python	JSON
Dict	Object
List	Array
Unicode	String
number - int, long	number – int
Float	number – real
True	True
False	False
None	Null

Changing Python information over to JSON is called an Encoding activity. Encoding is finished with the assistance of JSON library technique – dumps()

dumps() technique changes over word reference object of python into JSON string information design.

JSON to Python (Decoding)

JSON string interpreting is finished with the assistance of inbuilt strategy loads() and burden() of JSON library in Python. Here interpretation table show illustration of JSON objects to Python objects which are useful to perform unraveling in Python of JSON string.

Table 4.5 Json objects(Decoding)

JSON	Python
Object	Dict
Array	List
String	Unicode
number – int	number - int, long
number – real	Float
True	True
False	False
Null	None

PYGAME:

Python is the most mainstream programming language or nothing incorrectly to say that it is the cutting edge programming language. In each arising field in software engineering, Python makes its essence effectively. Python has tremendous libraries for different fields, for example, **Machine Learning (Numpy, Pandas, Matplotlib), Artificial knowledge (Pytorch, TensorFlow), and Game turn of events (Pygame,Pyglet)**.

In this instructional exercise, we will find out about game improvement utilizing the Pygame (Python library)

Pygame

- Pygame is a cross-stage set of Python modules which is utilized to make computer games.
- It comprises of PC illustrations and sound libraries intended to be utilized with the Python programming language.
- Pygame was authoritatively composed by Pete Shinners to supplant PySDL.
- Pygame is reasonable to make customer side applications that can be conceivably enclosed by an independent executable.

Pygame Installation

Introduce pygame in Windows

Prior to introducing Pygame, Python ought to be introduced in the framework, and it is a great idea to have 3.6.1 or above adaptation since it is a lot more amiable to amateurs, and also runs quicker. There are primarily two different ways to introduce Pygame, which are given underneath:

1. Introducing through pip: The pip instrument is a nice method to introduce Pygame (which is the thing that python uses to introduce bundles). The order is the accompanying:

```
py - m pip introduce - U pygame - client
```

2. Introducing through an IDE: The subsequent route is to introduce it through an IDE and here we are utilizing Pycharm IDE. Establishment of pygame in the pycharm is clear. We can introduce it by composing the order above into the terminal or utilize the accompanying advances:

o Open the File tab and snap on the Settings alternative

- .

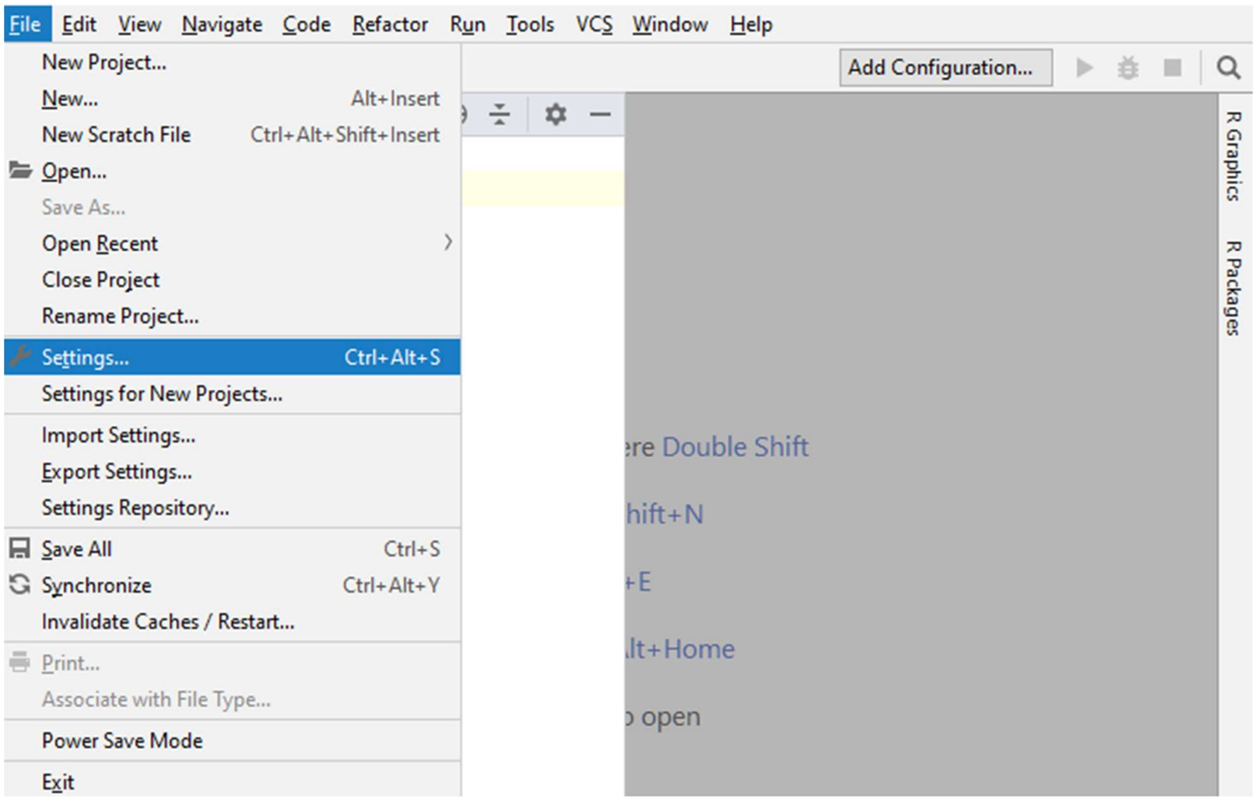
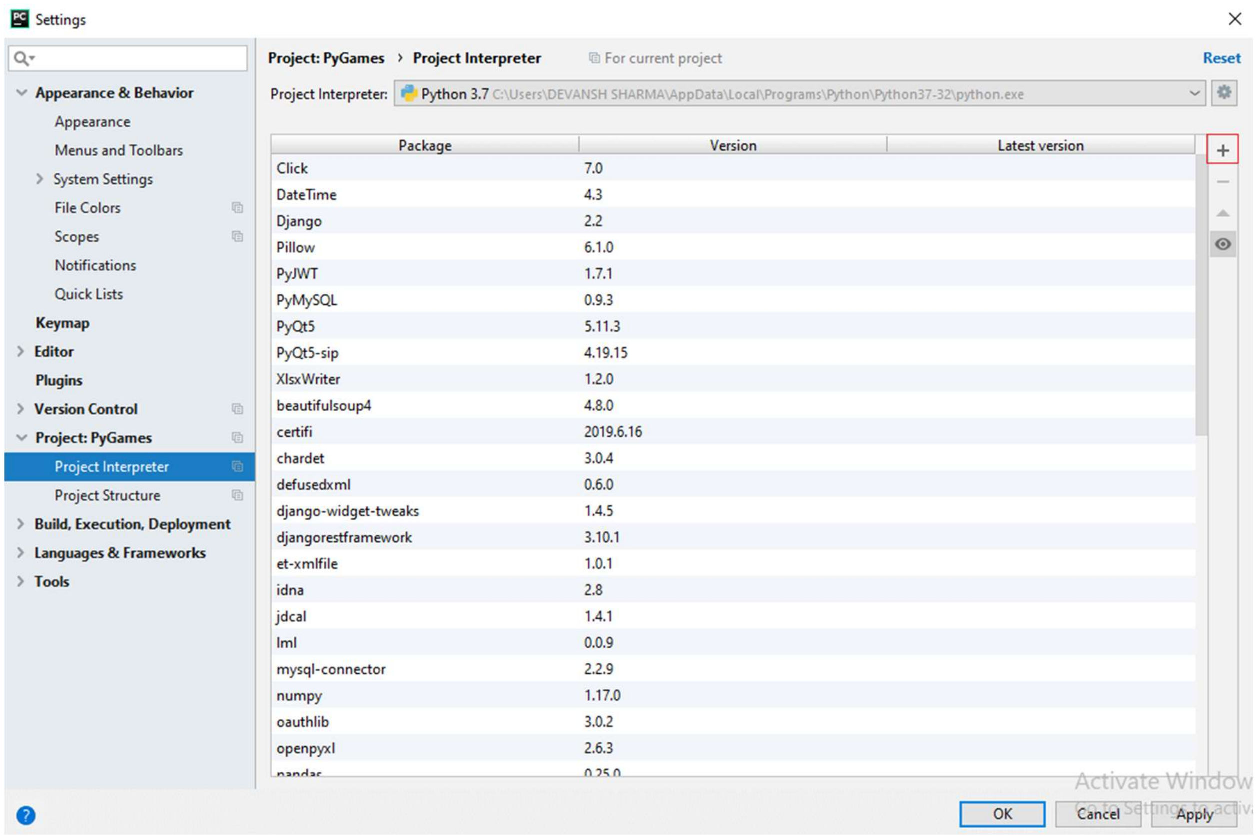


Fig 4.6 display of settings

- Select the **Project Interpreter** and click on the **+** icon.



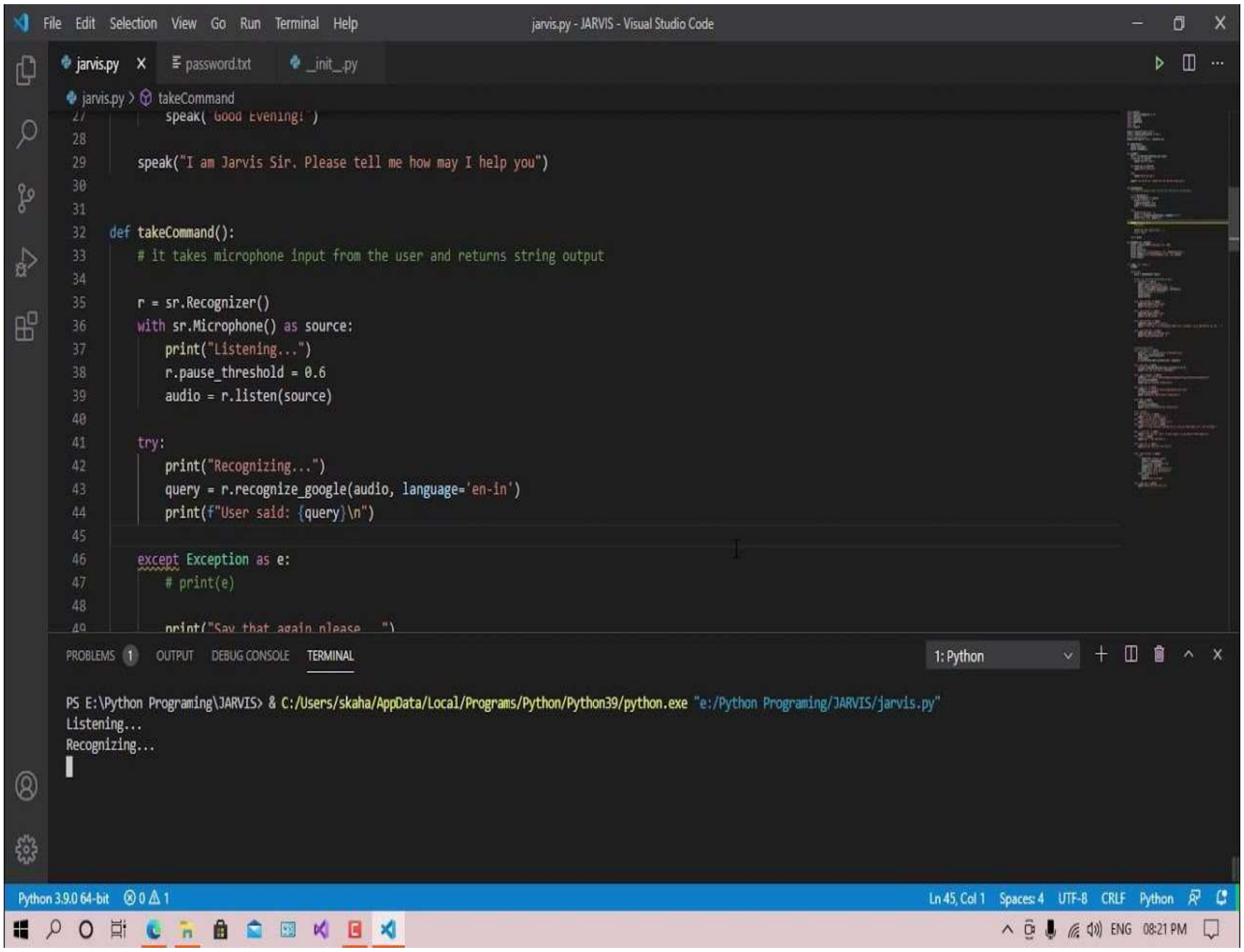
Project Interpreter: Python 3.7 C:\Users\DEVANSH SHARMA\AppData\Local\Programs\Python\Python37-32\python.exe

Package	Version	Latest version
Click	7.0	
DateTime	4.3	
Django	2.2	
Pillow	6.1.0	
PyJWT	1.7.1	
PyMySQL	0.9.3	
PyQt5	5.11.3	
PyQt5-sip	4.19.15	
XlsxWriter	1.2.0	
beautifulsoup4	4.8.0	
certifi	2019.6.16	
chardet	3.0.4	
defusedxml	0.6.0	
django-widget-tweaks	1.4.5	
djangorestframework	3.10.1	
et-xmlfile	1.0.1	
idna	2.8	
jdcal	1.4.1	
lml	0.0.9	
mysql-connector	2.2.9	
numpy	1.17.0	
oauthlib	3.0.2	
openpyxl	2.6.3	
pandas	0.25.0	

Fig 4.7 list of interpreters

It will display the search box. Search the pygame and click on the **install package** button.

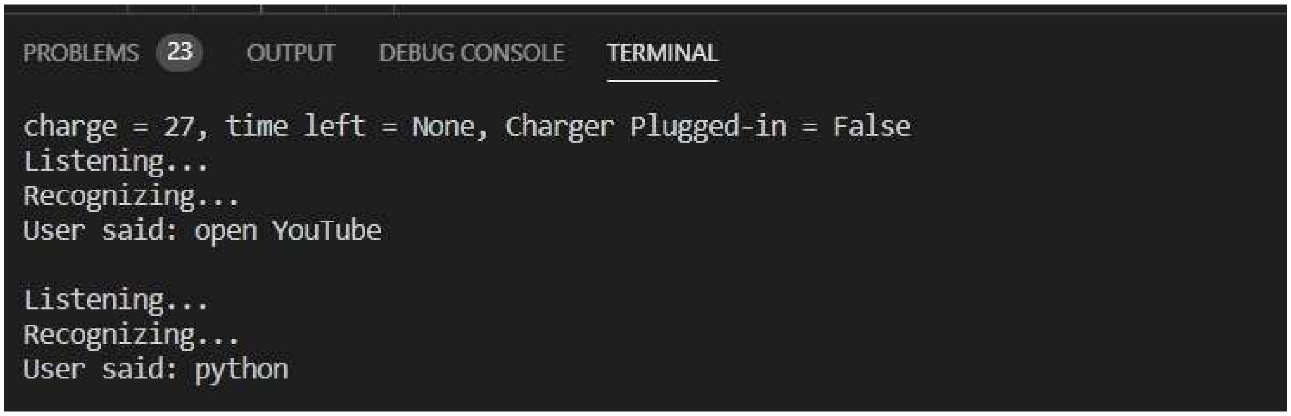
5 .Results and Analysis


File Edit Selection View Go Run Terminal Help
jarvis.py - JARVIS - Visual Studio Code
jarvis.py
password.txt
init.py
jarvis.py > takeCommand
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL
1: Python
Python 3.9.0 64-bit
Ln 45, Col 1 Spaces: 4 UTF-8 CRLF Python
ENG 08:21 PM

```python
        speak("Good Evening!")

    speak("I am Jarvis Sir. Please tell me 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:
        print("Listening...")
        r.pause_threshold = 0.6
        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...")
```

```
PS E:\Python Programing\JARVIS> & C:/Users/skaha/AppData/Local/Programs/Python/Python39/python.exe "e:/Python Programing/JARVIS/jarvis.py"
Listening...
Recognizing...
```

Fig. 5.1 running of program



```
charge = 27, time left = None, Charger Plugged-in = False
Listening...
Recognizing...
User said: open YouTube

Listening...
Recognizing...
User said: python
```

Fig. 5.2 Recognizing and processing youtube

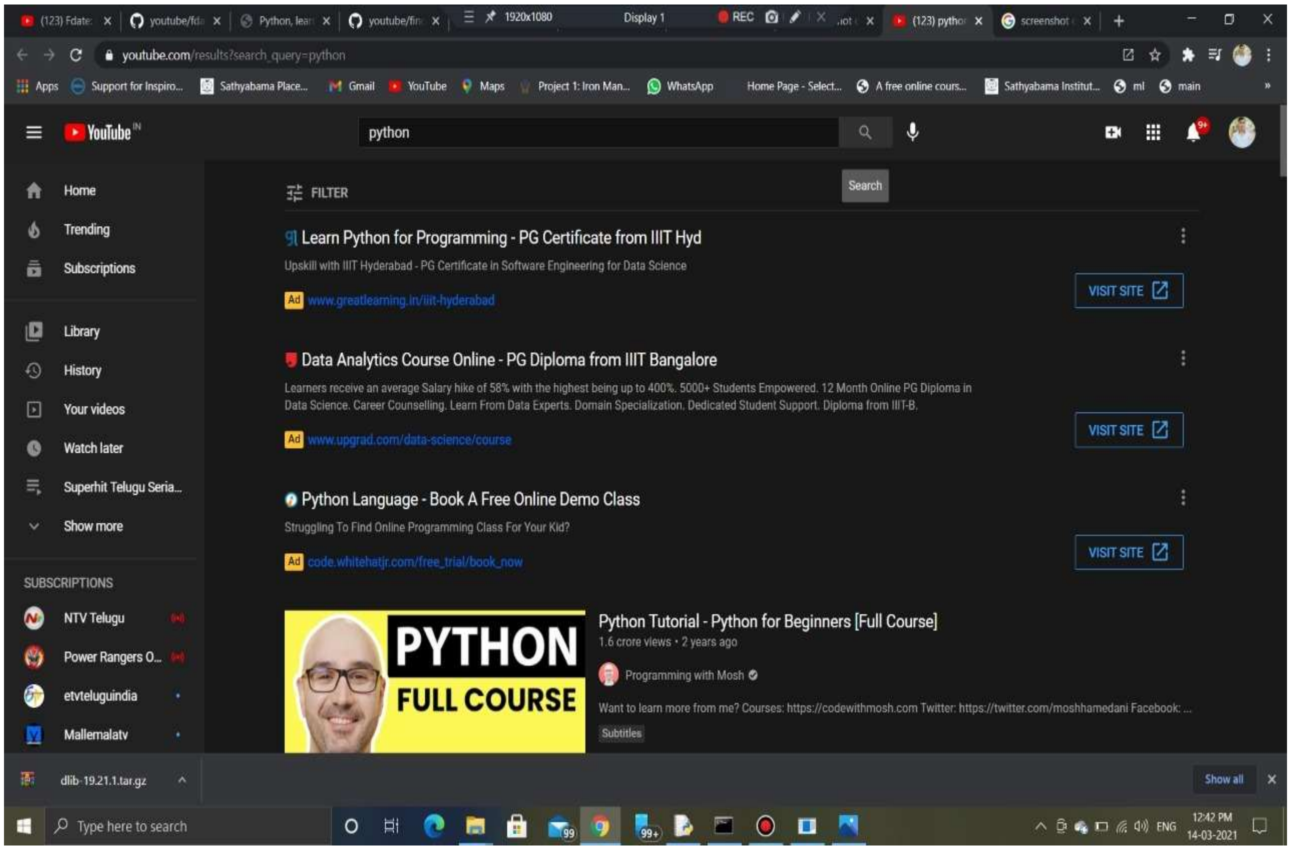
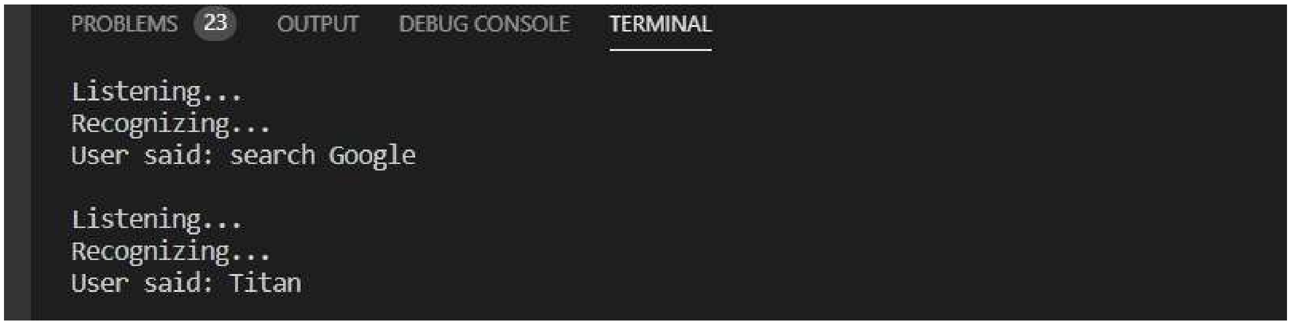
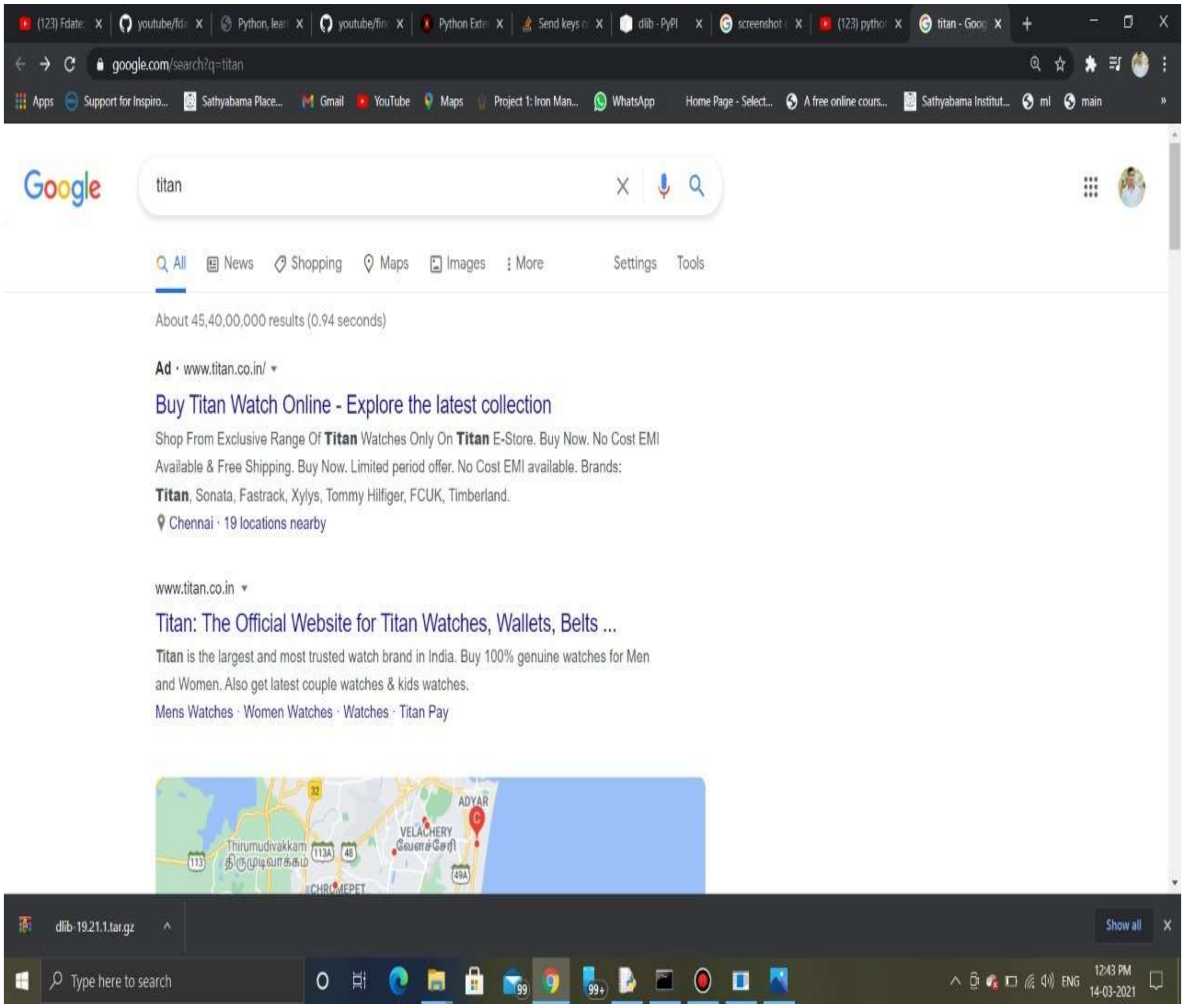

youtube.com/results?search_query=python
YouTube
python
FILTER
Learn Python for Programming - PG Certificate from IIIT Hyd
Data Analytics Course Online - PG Diploma from IIIT Bangalore
Python Language - Book A Free Online Demo Class
Python Tutorial - Python for Beginners [Full Course]
PYTHON
FULL COURSE
VISIT SITE

Fig. 5.3 Opened and searched result in youtube



```
Listening...
Recognizing...
User said: search Google

Listening...
Recognizing...
User said: Titan
```

Fig. 5.4 Recognizing and processing google search



About 45,40,00,000 results (0.94 seconds)

Ad · www.titan.co.in/

Buy Titan Watch Online - Explore the latest collection

Shop From Exclusive Range Of **Titan** Watches Only On **Titan** E-Store. Buy Now. No Cost EMI Available & Free Shipping. Buy Now. Limited period offer. No Cost EMI available. Brands: **Titan**, Sonata, Fastrack, Xylys, Tommy Hilfiger, FCUK, Timberland.

Chennai · 19 locations nearby

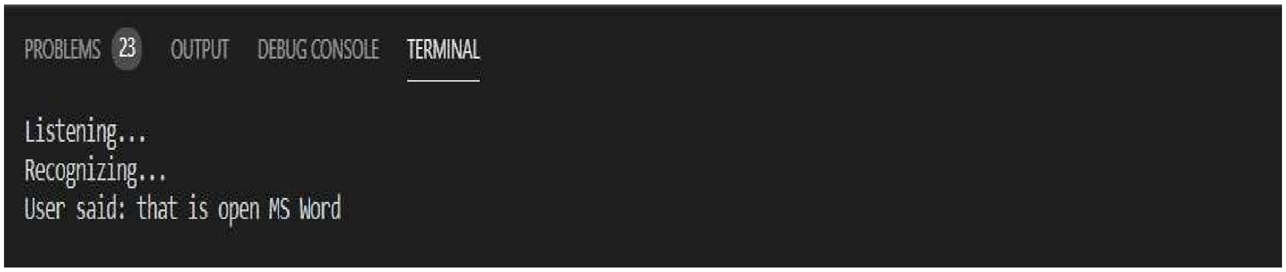
www.titan.co.in

Titan: The Official Website for Titan Watches, Wallets, Belts ...

Titan is the largest and most trusted watch brand in India. Buy 100% genuine watches for Men and Women. Also get latest couple watches & kids watches.

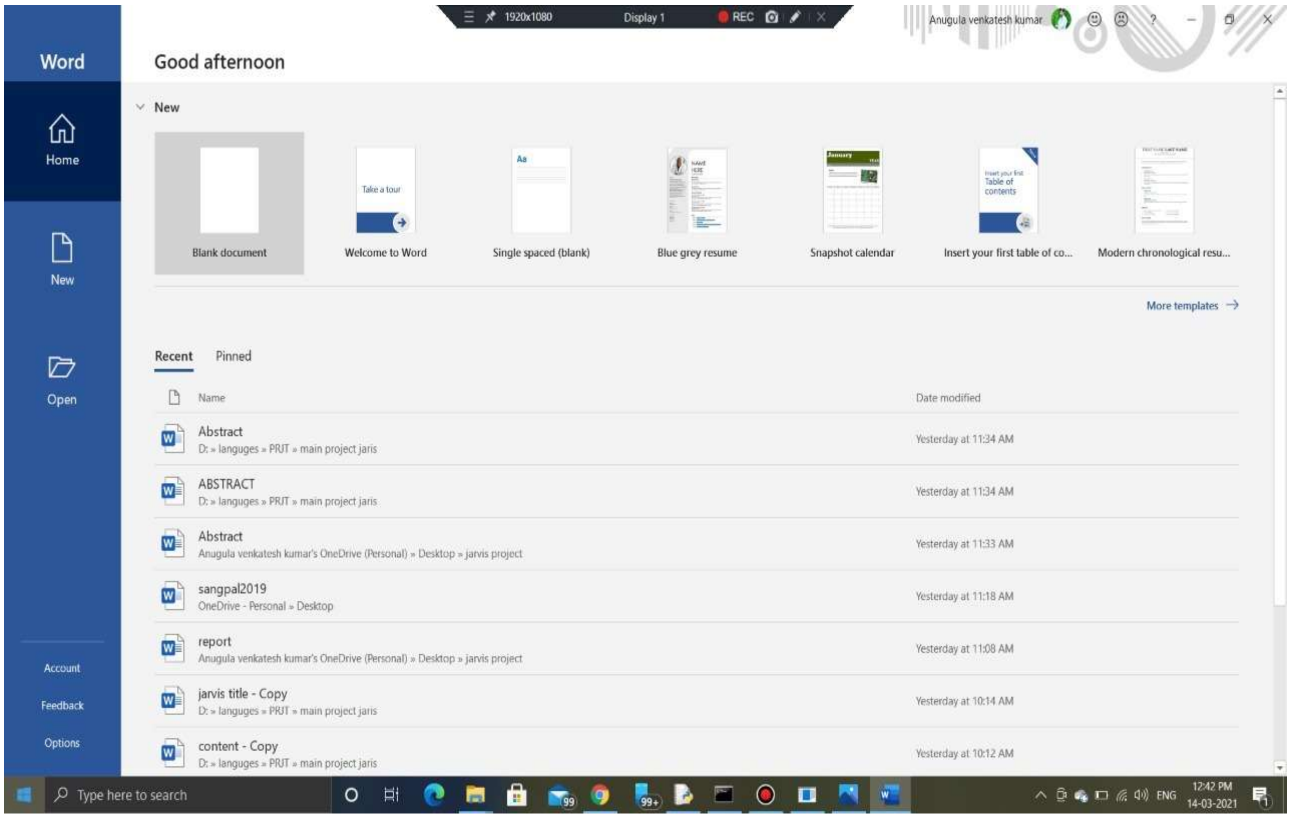
Mens Watches · Women Watches · Watches · Titan Pay

Fig. 5.5 Opened and searched result in google



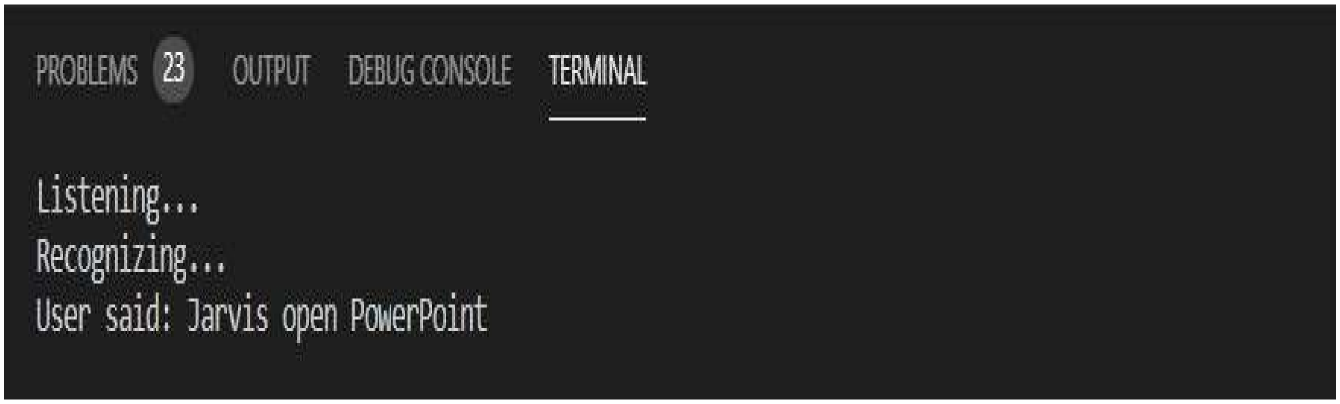
```
Listening...
Recognizing...
User said: that is open MS Word
```

Fig. 5.6 Recognizing and processing for MS word



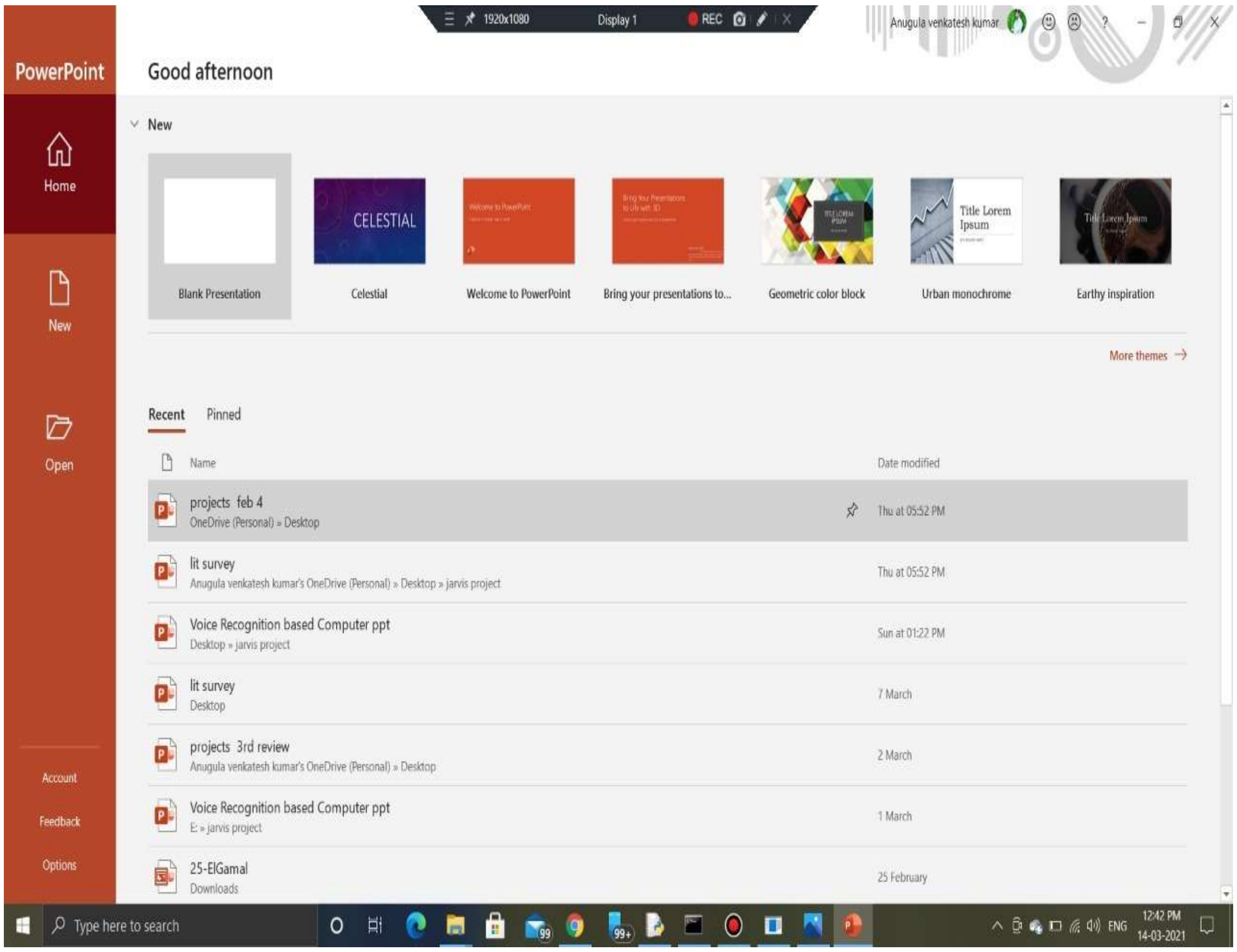
Name	Date modified
Abstract D: » languges » PRJT » main project jaris	Yesterday at 11:34 AM
ABSTRACT D: » languges » PRJT » main project jaris	Yesterday at 11:34 AM
Abstract Anugula venkatesh kumar's OneDrive (Personal) » Desktop » jarvis project	Yesterday at 11:33 AM
sangpal2019 OneDrive - Personal » Desktop	Yesterday at 11:18 AM
report Anugula venkatesh kumar's OneDrive (Personal) » Desktop » jarvis project	Yesterday at 11:08 AM
jarvis title - Copy D: » languges » PRJT » main project jaris	Yesterday at 10:14 AM
content - Copy D: » languges » PRJT » main project jaris	Yesterday at 10:12 AM

Fig. 5.7 Opened and searched result in MS word



```
Listening...
Recognizing...
User said: Jarvis open PowerPoint
```

Fig. 5.8 Recognizing and processing for Powerpoint



Name	Date modified
projects feb 4 OneDrive (Personal) » Desktop	Thu at 05:52 PM
lit survey Anugula venkatesh kumar's OneDrive (Personal) » Desktop » jarvis project	Thu at 05:52 PM
Voice Recognition based Computer ppt Desktop » jarvis project	Sun at 01:22 PM
lit survey Desktop	7 March
projects 3rd review Anugula venkatesh kumar's OneDrive (Personal) » Desktop	2 March
Voice Recognition based Computer ppt E: » jarvis project	1 March
25-ElGamal Downloads	25 February

Fig. 5.9 Opened and searched result in MS powerpoint

Some examples :

Tell me the breaking news

Play a music

Tell me weather report

Send gmail

Alarm

Shutdown the system

Restart

Sleep …

Stop the service

Go offline/ go to sleep / exit / quit / bye / goodbye

CHAPTER 6

CONCLUSIONS & FUTURE WORK

6.1. CONCLUSION

In our project we have implemented many things compared to other assistants. Now a days it is very useful in human life because it is a hands-free application. It is a very simple application. As well as it is used in a business field also for example in laboratory, the person wears gloves and body suits for their safety purpose so it is difficult to type, through voice assistant they can get any information so that their work becomes easy.

Voice assistants are useful in many fields such as education, daily life application, home appliances etc. and voice assistant is also useful for the illiterate people they can get any information just by saying to the assistant, luxury is available for people, thanks to AI based voice assistants. Voice assistant is developing more and more in daily life. Many companies of voice assistant trying to improve interaction and more features to the next level and many of the youth started using voice assistant in daily life and from many sources the result showing very good feedback. Compared to last 5 years voice assistants have been developed more and more

6.2. FUTURE WORK

The future work for our project is very interesting such that it may become more popular than the present thing if it was implemented. The future work is quite different and interesting. It is like adding hand motion gesture to the present thing which leads to very advanced level. By the movement of the hand, we can control our system/Laptop. Each motion is given a particular instruction such that by the motion of the hand it performs the task that usually taken by voice command. This is very much interesting and easy to control.

REFERENCES

[1]. Artificial intelligence (AI), sometimes called machine intelligence. https://en.wikipedia.org/wiki/Artificial_intelligence

[2]. Bohouta.G, Këpuska V.Z, "Comparing Speech Recognition Systems (Microsoft API Google API And CMU Sphinx)", Int. Journal of Engineering Research and Application 2017, 2017

[3]. CMUSphnix Basic concepts of speech - Speech Recognition process. http://cmusphinx.sourceforge.netlwiki/tutorialconcepts

[4]. Cortana Intelligence, Google Assistant, Apple Siri.

[5]. Fryer, L.K. and Carpenter, R., 2006. Bots as tools for language learning Language Learning & Technology

[6]. Hill, J., Ford, W.R. and Farreras, I.G., 2015. A study of human–human online conversations and human–chatbot conversations using artificial intelligence in real life. Human Behaviour with Computers

[7]. Huang, J., Zhou, M. and Yang, D., 2007, January. Extracting Chatbot Online Discussions Forums Provide Information. In IJCAI(Vol. 7, pp. 423-428).

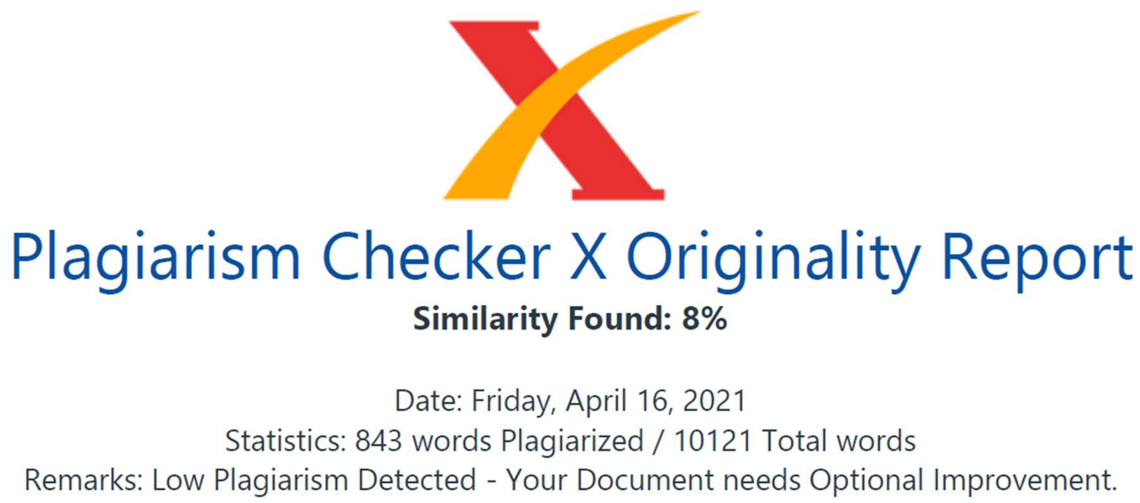
[8]. Marr. B, The Amazing Ways Google Uses Deep Learning AI.

[9]. Mohasi, L. and Mashao, D., 2006. Text-to-Speech Technology in Human-Computer Interaction. In 5th Conference on Human Computer Interaction in Southern Africa, South Africa (CHISA 2006, ACM SIGHI) (pp. 79-84).

[10]. Noda.K, Arie.H, Suga.Y, Ogata.T, Multimodal integration learning of robot behavior using deep neural networks, Elsevier: Robotics and Autonomous Systems, 2014.

[11]. Thakur, N., Hiwrale, A., Selote, S., Shinde, A. and Mahakalkar, N., Artificially Intelligent Chatbot.

PLAGIARISM



Plagiarism Checker X Originality Report

Similarity Found: 8%

Date: Friday, April 16, 2021

Statistics: 843 words Plagiarized / 10121 Total words

Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

INTERNET SOURCES:

<1% - https://www.cr-report.telekom.com/site20/

<1% - https://www.academia.edu/41278235/ISE_690_Cyber_Security_Capstone

<1% - https://blog.robosoftin.com/author/bradley-james/

<1% - https://www.techiexpert.com/understanding-the-convergence-of-iot-and-data-analytics/

<1% - https://patents.google.com/patent/US20040030556A1/en

<1% - https://www.tumbral.com/tag/voice%20assistant

<1% - https://link.springer.com/chapter/10.1007/978-3-030-08277-2_8

<1% - https://www.ml.cmu.edu/cmsint/mldcmu.rss

<1% -
https://www.researchgate.net/publication/306063238_Smart_answering_Chatbot_based_on_OCR_and_Overgenerating_Transformations_and_Ranking
<1% - https://www.researchgate.net/publication/327391803_A_Knowledge-based_Methodology_for_Building_a_Conversational_Chatbot_as_an_Intelligent_Tutor
<1% - https://www.irjet.net/archives/V7/i10/IRJET-V7I10164.pdf
<1% - https://issuu.com/cdtm/docs/siemens_ai_final_report_compressed
<1% - https://www.aclweb.org/anthology/volumes/D19-1/
<1% - https://digitalonebox.com/standard-blog/
<1% - https://www.analyticsinsight.net/complete-guide-natural-language-processing-nlp/
<1% - https://www.myprivatetutor.ae/training-center/1512
<1% - https://www.qtreetechnologies.in/course/python-training-in-coimbatore.php
<1% - https://placementps.com/python-training-in-chennai/
<1% - https://www.acte.in/data-structures-with-python-cheat-sheet-tutorial/
<1% - https://ukdiss.com/examples/pattern-prediction-geo-spatial-data-analysis.php
<1% - https://pythonupodates.blogspot.com/
<1% - https://veteranlogix.com/roadmap-of-back-end-development-2020/
<1% - https://www.scribd.com/document/311790036/Python
<1% - https://www.dasinfomedia.com/services/web-development/hire-dedicated-python-developer/
<1% - https://play.google.com/store/books?hl=fr&gl=US
<1% -
https://www.academia.edu/43654719/Yves_Hilpisch_Python_for_Finance_Mastering_Data_Driven_Finance_SECOND_EDITION
<1% - https://ascl.net/code/all/limit/2097/dir/desc/order/title/listmode/full
<1% - https://blog.csdn.net/cunzai1985/article/details/108752545
<1% - http://crowdforthink.com/rss
<1% - https://hackthedeveloper.com/python-subprocess-execute-shell-commands/
<1% - https://documentation.help/Python-3.5/subprocess.html
<1% - https://link.springer.com/article/10.1007/s42452-020-03870-0
<1% - https://www.teckmart.com/text-to-speech-online-mp3-voice-2021/
<1% - https://www.techtravelhub.com/nosuchsessionexception-in-selenium/
<1% - https://ttsoffline.wordpress.com/2016/06/
1% - https://pyttsx3.readthedocs.io/_/downloads/en/stable/pdf/
<1% -
https://github.com/RapidWareTech/pyttsx/commit/6fd0b8871e661d5d1caff9aac075cfda9116dc23

1% - https://pyttsx3.readthedocs.io/en/latest/engine.html
<1% - https://www.assignmentessays.com/
<1% - https://pyttsx.readthedocs.io/en/v1.0/engine.html
<1% - https://github.com/nateshmbhat/pyttsx3/issues/78
1% - https://bbs.huaweicloud.com/blogs/115130
1% - https://github.com/nateshmbhat/pyttsx3/blob/master/docs/engine.rst
<1% - https://sharpcommunication.blogspot.com/
<1% - https://www.guru99.com/python-json.html
<1% - https://ufdc.ufl.edu/AA00016616/00020
<1% - https://cag.gov.in/uploads/download_audit_report/2017/Chapter_6_-_Conclusion_and_Recommendations_of_Report_No.22_of_2017_-_Performance_audit_Union_Government_Electrification_Projects_Reports_of_Indian_Railways.pdf
<1% - https://www.motivation2learn.com/
<1% - https://dl.acm.org/doi/10.1145/3415223