

VOICE ASSISTANT USING PYTHON

¹A.Venkata Divya Sri, ²Mr.P.Rameshbabu

¹(Student Dept. of Master of Computer Applications, Amrita Sai Institute of Science and Technology, Paritala, Andhra Pradesh, 521180, India.

Email: allikapaldivyasri@gmail.com)

²(Asst.Prof, Dept. of Computer Science & Engineering, Amrita Sai Institute of Science and Technology, Paritala, Andhra Pradesh, 521180, India.

Email: asistithod@gmail.com)

Abstract:

A voice assistant that simplifies routine work through voice commands has been developed using Python in this project. The assistant translates user speech to text, interprets it, responds with sound and writes out text as well by using speech recognition, natural language processing and text-to-speech technology. It answers questions, weathers cheques, shows you time, opens apps and handles web searches, all by itself. Using SpeechRecognition, pyttsx3 and datetime libraries, the system shows how Python can easily develop an assistant that customers can use with ease. To ensure it can change and upgrade, the design uses live interaction, has a modular setup and is extendable now and in the future. Thanks to the assistant, users having trouble with sight or who are using their hands can access the web more easily and developers can add new features less laboriously. Using Python and a number of Python tools, the project outlines how to build a functional AI-powered assistant and forms the basis for further progress in deep learning, voice recognition and running applications across many platforms. The overall result reflects the trend toward human-computer interaction that depends on voice, personalised settings and being aware of the user's context.

Keywords — *Voice Assistant, Python, Speech Recognition, Natural Language Processing, Text-to-Speech, Human-Computer Interaction.*

1. Introduction

Voice assistants are now important for communicating with technology, as they provide an easy and quick way to manage your tasks, find data and manage devices around you. As technology improves, voice recognition is becoming the best way to have easier interactions. As new research and development is interested in building personalised, special-purpose assistants, interest has grown around popular voice assistants on the market like Siri, Alexa and Google Assistant. Machine learning and automation are powerfully demonstrated by these technologies.

With this project, we work to replicate a basic virtual assistant using Python. By using the

speech_recognition and pyttsx3 libraries, it hears what the user says, works on the request and replies with the correct behaviour. Since you can open applications, cheque the time and find information hands-free, the assistant works well for activities in various areas.

Such a project finds Python very well suited because it has easy-to-use code and lots of useful libraries. Communication input is constantly monitored, processed and actions are taken to assure a perfect and immediate experience. Even though the assistant's NLU is not advanced, it is in place now and can be built on later with new options such as machine learning, OpenAI's GPT or working with IoT devices.

Secret and safe browsing are also taken into account because with the assistant on your machine, your privacy isn't a concern because it doesn't require an online connexion. The project shows how AI, talking with devices, automation and system integration work in practise.

Design:

- Users say "Hello NOVA" to activate the assistant.
- User speech is recognised and written down by the assistant before the system figures out what it means.
- Assistant provides an answer according to the keywords used by the user. If the request returns nothing, users are urged to cheque again.
- The system communicates by synthesising speech.

Voice Assistant - NOVA :

Similar to commercial voice assistants, NOVA allows users to control tasks and get information using simply their voice. Because the assistant requires only a simple command, uses voice feedback and answers quickly, tasks can be accomplished more easily by people with disabilities, as well as when many things are happening at once. Because more people are using voice search, NOVA functions as an excellent product for enhancing regular dealings with technology.

2. Problem Statement

Because of recent rapid improvements in AI and ML, digital assistants controlled by voice are now a common form of human computing. Because these assistants perform tasks with voice commands, they help users access information from a distance, so there is less reason to touch devices. The goal of this paper is to show how a Python-built voice assistant performs the same tasks as a real digital

assistant, with a major focus on ease of use and talking with hands free.

Our main aim is to set up a voice assistant that makes it simpler to cheque the weather, search the internet, set alerts, email from anywhere and control smart home devices. In order to make the experience easy to use, the assistant tries to combine speech recognition, NLP and TTS features.

The first main part of a voice assistant is speech recognition which turns things you say into a written record. If the process works accurately, it gathers speech of many types, like different accents, tones and talking speeds. Thanks to SpeechRecognition and PyAudio, speech input can be trusted and will work properly in a range of environments.

After the voice is converted, the job of the assistant is to grasp the user's intent. NNP handles this by examining the transcribed speech to understand the hidden message, things the user is telling it to do and the objects in question. NLTK, spaCy and creating your own set of rules can all help the assistant answer as needed.

A main advantage of these tools is that they can respond to users via speech. Pyttsx3 and gTTS offer a TTS function that turns the assistant's messages into human speech, improving communication and simple use of the application.

Because of this, the job of the assistant includes communication with diverse data sources and APIs. To give answers to weather questions, the system uses services such as OpenWeatherMap. In addition, to remind you, your assistant connects with the timing functions of the operating system. Because Microservices are modular, it is easy to add extra features or commands and continue with existing services without any disruption.

Achieving success with a voice assistant comes from quick, proper and time-efficient user experience. Making sure errors are handled and performance will be good secures the reliability of your dataset.

Protection of both privacy and security comes first when working with confidential data. In the ideal case, the system will deal with data directly on a local device to avoid send it over the internet.

A lot of noise around a speaker and acoustic sounds may affect how speech is recognized, but you can improve recognition in loud areas using silence-embedding audio systems and special microphones.

If the platform personalises itself based on users' actions and gives support in multiple languages, more people can find it interesting and simple to use. Alternative to using their voices, users can use a graphical user interface (GUI) to enhance their interaction.

The assistant's dependability is guaranteed when many tests, including unit and usability tests, are carried out. Applying user comments can improve the operations of the system. Build your architecture according to tidy coding standards and ensure your documentation is clear if you work on an open-source project.

Because every platform (desktop, mobile or Raspberry Pi) has its specific limitations, their deployment should be planned for. Adding smart home devices allows the assistant to become your home's automatic control hub.

Commands you haven't taught your assistant should lead to brief prompts for clarification. These tools are used to watch how a website performs and to guide improvement steps.

This project links AI, machine learning, voice recognition and software engineering, allowing people to use and understand voice assistance.

3. Objective:

The project will create a Voice Assistant that can detect spoken instructions using Python, carry out tasks such as telling what time it is, launching applications or web pages, running web searches, sending emails and answering with synthesised sound. The assistant obtains the ability for smooth

interaction by including libraries such as `speech_recognition` and `pyttsx3`, as well as tools like `os`, `webbrowser` and `datetime`.

Because you can give commands by voice, the system makes it easier and more convenient for anyone, especially if their hands are full or no other input is available. The project acts as a tool for practical use and also functions as a learning resource, teaching speech processing and natural language understanding and demonstrating their use in practise.

Another target is to show how speaking and writing software can be used to make a practical assistant even more useful. It provides a basis for improving smart technologies and personal virtual assistants in the future.

Moreover, since a project supports its own enhancements, people can use voice authentication, receive personal responses and plug in APIs so they know about the weather. As a result, this tool lets you practise Python, automate systems and explore the fast growth of voice technology, so it's both useful and easy for anyone to learn.

4. Importance and Applications

4.1 Importance

This project shows how voice-controlled technology can be used to improve how users interact with programmes. It shows that Python can help in constructing straightforward voice applications that improve accessibility and support doing more work at once for everyone, especially those with special needs or overloaded schedules. In this project, you get a basic understanding of speech recognition, natural language processing and text-to-speech, all while using the tools `speech_recognition` and `pyttsx3`. It also answers the rising interest in voice applications used in smart homes, medical care and cars.

4.2 Applications

Using Python, the Voice Assistant project supports many areas where it enhances productivity, accessibility and user enjoyment.

- Using Personal Productivity, you can set up reminders, check what the weather is like, manage your emails and organise events.
- With home automation, it is possible to turn lights, thermostats and security cameras on and off simply by speaking.
- Easily accessible: Give individuals who have trouble with physical movements a new way to use computers or smart devices, so they don't have to use a keyboard or mouse.
- Using voice commands or by searching online, you can fast access both news updates and weather forecasts, as well as answers to specific questions.
- You can support a student's learning, language work or research through voice-based information delivery.
- With these applications, medical workers can make and manage appointments, gather patient's information and execute different tasks by talking into the device.
- By using e-commerce, you can easily find products, check prices and complete purchases using your voice.
- Control musicians, actors and podcasters with your voice by telling your device what you want to listen to.
- Handle IoT gadgets in your home to enjoy greater intelligence, energy savings and safety.
- You should be able to help customers resolve basic issues, handle troubleshooting

and supply automated support in any business setting.

5. Requirements And Specifications

5.1 Hardware Configuration

Component	Client Side	Server Side
RAM	1GB	1GB
Hard Disk	10GB	20GB
Processor	Not specified	2.4GHz

TABLE1: HARDWARE-CONF

5.2 Software Requirements

Component	Client Side	Server Side
Operating System	Not specified	Windows 11
Web Browser	Google Chrome or any compatible browser	Google Chrome or any compatible browser
Server-side Language	Not specified	Python

TABLE2: SOFTWARE-REQ

5.3 System Architecture

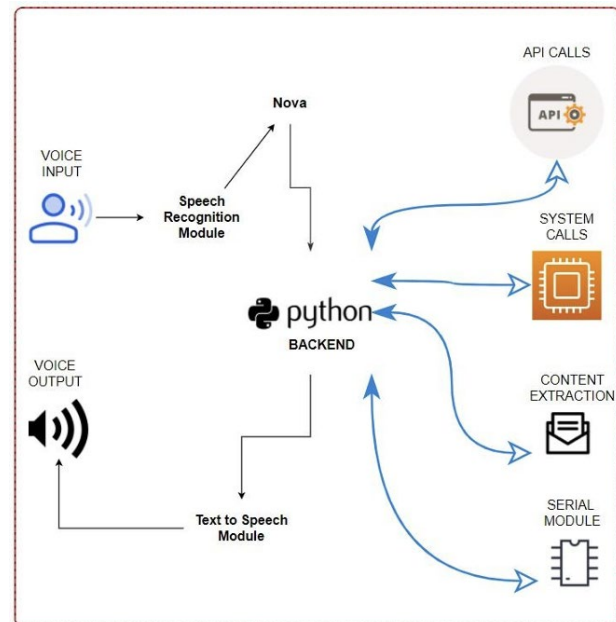


Fig1:Sys-Architecture

5.4 System Design

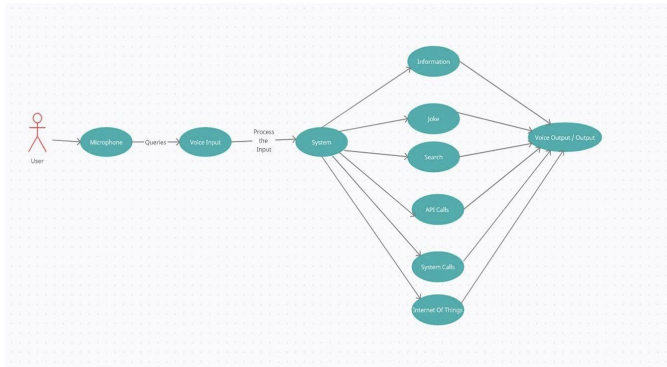


Fig2:Sys-Design

5.5 Sequence Diagram

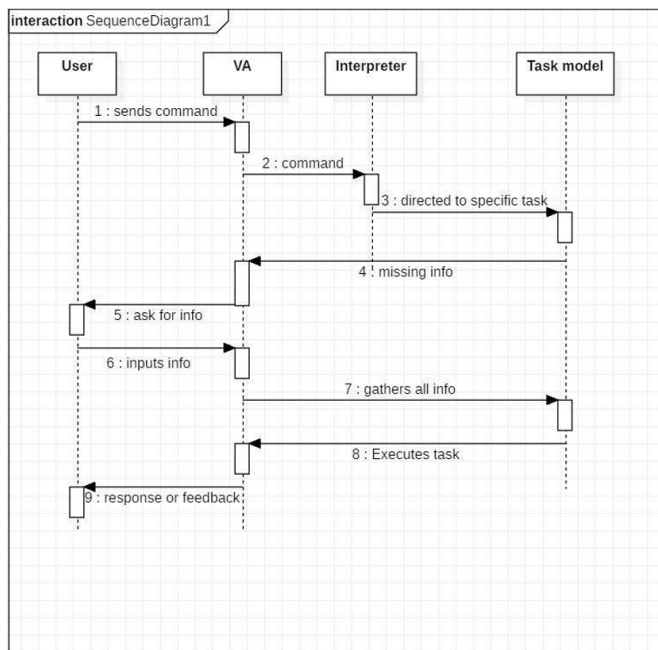


Fig3:Sequence-Diagram

5.6 ALGORITHMS USED

1. As part of Automatic Speech Recognition, this module recognises the spoken word.

- The programme uses the Recognizer class to create text from spoken audio and gives verbal feedback.
- The `energy_threshold` function serves to distinguish when sound is speech, vs. when it is silence.

- `recognizer.adjust_for_ambient_noise` (source, duration=1) says the model how loud the background noise is using the source's audio.

2. Technologies That Allow Us to Turn Words into Sound and Sound into Words.

- `pyttsx3` allows you to convert any text to speech and you can customise the voice, rate and volume.
- The Speech Recognition API turns sound into text so the result can be used for other purposes.

- ChatBot makes voice synthesis available with SAPI5 and eSpeak.

3. A gateway executes both commands and dictates their outcomes.

- All user instructions are taken as language by Speech Recognition and held in temporary storage.
- The command is studied and the data in the input specifies which action the system should take and when to execute it.

5.7 Dataset

For a Python-based voice assistant to work well, it needs a reasonable dataset. If that is necessary for the project, the datasets might be composed of speech commands, recordings of speech or exchanged text. You can find a list of public datasets below, together with a sample one in Python format, to help with voice application development.

1. Google Speech Commands Dataset

- URL: https://www.tensorflow.org/datasets/catalog/speech_commands

- **Description:** The data includes 65,000 recordings of one-second clips for 30 different short words. Recognising certain words or phrases is one of the most useful things it does, for example, words and phrases such as, "yes," "no," and "stop."

2. Mozilla CommonVoice

- **URL:** <https://commonvoice.mozilla.org/en/datasets>
- **Description:** This collection, Mozilla CommonVoice, contains lots of labelled voice data in various languages, designed for use in speech recognition and language modelling.

3. LibriSpeech

- **URL:** <http://www.openslr.org/12>
- **Description:** There are thousands of hours of English audio from audiobooks in the LibriSpeech collection. The dataset is very useful when developing speech recognition systems.

4. OpenSubtitles

- **URL:** <https://opus.nlpl.eu/OpenSubtitles.php>
- **Description:** OpenSubtitles is made up of a large collection of subtitles which are useful for creating chat-style voice assistants.

6. Output

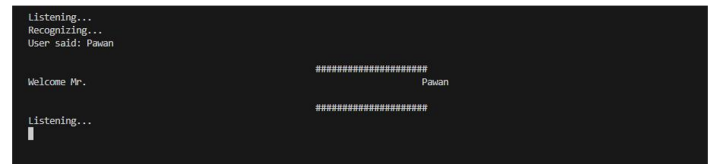


Fig4:Conversion

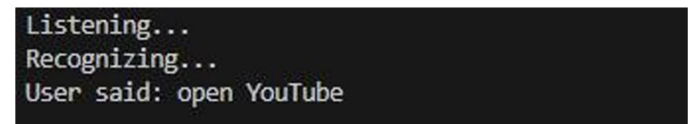


Fig5:Processing-Instructions

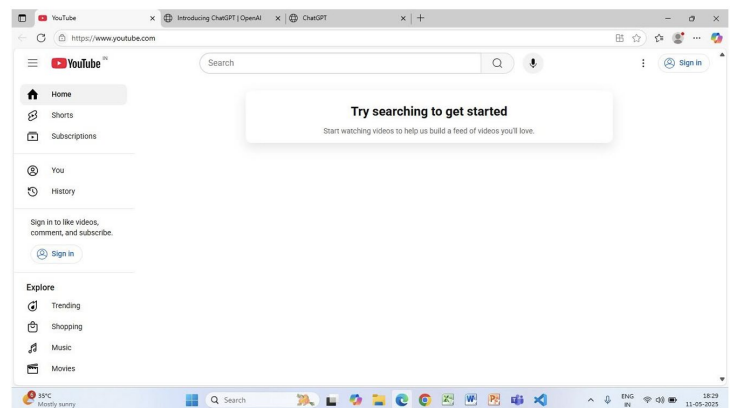


Fig5:Output

7. Conclusion

Anyone looking to build a voice assistant with Python will find it encouraging, since it reveals the abilities of modern programming to improve communication between people and machines. Python's clear syntax and helpful libraries made connecting speech recognition (through `speech_recognition`), text-to-speech (`pyttsx3`) and several other useful services possible, leaving the programme easy to use. In the project, tackling background noise and speech clarity meant it became clear how important error management and being adaptable really are.

Usability was a main priority in the assistant's creation, ensuring it gives access to all and works well. Practical tools like time, application opening and browsing the web were included, along with benefits for

users with disabilities. In addition, this project provided a base for further studying NLU and machine learning, elements that can greatly improve the assistant.

Acknowledgement

I am thankful to the Management of Amrita Sai Institute of Science and Technology for giving me an opportunity to work with his project.

I would like to thank **Dr. M. Sasidhar**, Principal, Amrita Sai institute of science and technology, for his constant encouragement and support during the progress of this work.

I am deeply grateful to **Dr. P. Chiranjeevi**, Professor and Head of the Department, for his valuable guidance and consistent support during the course of the project.

A special note of thanks to my internal guide, **Mr.P.Rameshbabu (M.Tech)**, for his exceptional guidance, constant motivation, and continuous encouragement, which played a crucial role in the successful completion of this project.

A.VENKATA DIVYA SRI

References

- [1] Python Software Foundation, "Python 3.10 Documentation," 2023. [Online]. Available: <https://docs.python.org/3/>. [Accessed: May 21, 2025].
- [2] Uberi, "SpeechRecognition Library," GitHub Repository, 2021. [Online]. Available: https://github.com/Uberi/speech_recognition. [Accessed: May 21, 2025].
- [3] pyttsx3 Developers, "pyttsx3: Text-to-Speech Conversion Library in Python," 2022. [Online]. Available: <https://pyttsx3.readthedocs.io/>. [Accessed: May 21, 2025].
- [4] GeeksforGeeks, "Python Voice Assistant Using SpeechRecognition and pyttsx3," 2023. [Online]. Available: <https://www.geeksforgeeks.org/python-voice-assistant-using-speechrecognition-pyttsx3/>. [Accessed: May 21, 2025].
- [5] TutorialsPoint, "Python Speech Recognition Tutorial," 2022. [Online]. Available: <https://www.tutorialspoint.com/python/>. [Accessed: May 21, 2025].
- [6] W3Schools, "Python Tutorial," 2023. [Online]. Available: <https://www.w3schools.com/python/>. [Accessed: May 21, 2025].
- [7] Stack Overflow, "Various Discussions on Python Voice Assistant Errors and Enhancements," n.d. [Online]. Available: <https://stackoverflow.com/>. [Accessed: May 21, 2025].
- [8] RealPython, "Build a Python Command-Line Application," 2023. [Online]. Available: <https://realpython.com/>. [Accessed: May 21, 2025].
- [9] Microsoft Azure, "Speech Service Documentation," 2023. [Online]. Available: <https://learn.microsoft.com/en-us/azure/cognitive-services/speech-service/>. [Accessed: May 21, 2025].

- [10] Google Cloud, "Speech-to-Text API Documentation," 2023. [Online]. Available: <https://cloud.google.com/speech-to-text/docs>. [Accessed: May 21, 2025].
- [11] Mozilla, "Common Voice Dataset," 2022. [Online]. Available: <https://commonvoice.mozilla.org/>. [Accessed: May 21, 2025].
- [12] Medium, "How to Build Your Own Voice Assistant Using Python," n.d. [Online]. Available: <https://medium.com/>. [Accessed: May 21, 2025].
- [13] Towards Data Science, "Creating a Voice Assistant with Python and NLP," 2023. [Online]. Available: <https://towardsdatascience.com/>. [Accessed: May 21, 2025].
- [14] Postman, "API Collections for Developers," 2023. [Online]. Available: <https://www.postman.com/>. [Accessed: May 21, 2025].
- [15] RapidAPI, "Free APIs for Voice Assistant Projects," 2023. [Online]. Available: <https://rapidapi.com/>. [Accessed: May 21, 2025].
- [16] Tech with Tim, "Creating a Jarvis-like Voice Assistant in Python," YouTube Video, 2021. [Online]. Available: <https://www.youtube.com/>. [Accessed: May 21, 2025].
- [17] Kaggle, "NLP and Speech Recognition Datasets and Notebooks," 2023. [Online]. Available: <https://www.kaggle.com/>. [Accessed: May 21, 2025].
- [18] IEEE Xplore, "Research Papers on Voice Assistants and Human-Computer Interaction," n.d. [Online]. Available: <https://ieeexplore.ieee.org/>. [Accessed: May 21, 2025].
- [19] Berkman Klein Center for Internet & Society, "Privacy and Ethical Issues in Voice Technology," 2022. [Online]. Available: <https://cyber.harvard.edu/>. [Accessed: May 21, 2025].
- [20] Electronic Frontier Foundation (EFF), "Voice Assistants and Surveillance Concerns," 2022. [Online]. Available: <https://www.eff.org/>. [Accessed: May 21, 2025].