A Midterm Progress Report

on

DESKTOP ASSISTANT

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

Computer Science and Engineering

Submitted by

Dilpreet Kaur	Manreet Kaur	Arshdeep Singh
2104091	2104142	2104075

UNDER THE GUIDANCE OF

Er. Jasdeep Kaur

(August-2024)

Department of Computer Science and Engineering

GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA

Index

Content	Pages
1. Introduction 1.1 Detailed Introduction 1.2 Objectives	3-8
2. System Requirements	9-10
3. System Requirements Analysis	11-13
4. Software Design 4.1 design 4.2 Proposed Architecture 4.3 Proposed Approach 4.4 Working Model 4.5 Execution	14-19
5. Testing Module	20-24
6. Performance 6.1 Performance of project developed do far	25-28
7. Output Screens	29-31
8. References	32

1.INTRODUCTION

It is also a personal desktop assistant, which will help the end-user with most daily activities, for instance by having casual conversations, searching, looking up word definitions, finding information about medications, and giving health recommendations based on symptoms; besides, it can look up weather, open various desktop applications, such as FaceTime and browsers, based on verbal instruction, music, telling time, and providing weather updates based on user requests. It uses machine learning in the evaluating of commands and statements of the users for the best outcome. Today, almost everything is managed online. With this smartphone in our hands, literally the whole world is at our fingertips. We do not even use our fingers anymore. All we say about the work is done. There are applications that enable us to type in the words "machine learning" and the desktop assistant will give us the answer. The job of a virtual assistant is to do just that. The concept of this project is based on the fact that there is enough freely accessible data and information on the web in order to make a virtual desktop assistant that could have judgments for routine tasks by the users. Virtual assistant helps us save our time. Virtual assistant provides flexibility for a user to modify us functionalities. This virtual assistant to your computer has to go from basics python to complex programming accordingly. Virtual assistant has an ability to understand and perform requests. The virtual assistant generally offers a useful tool that streamlines and enhances the activities within the boundaries of computer-based activities, making it easier for them to get around with technology and accomplish their goals in quite an efficient and natural way. Artificial intelligence could enable natural conversations between humans and machines. Virtual assistants are an early version of such technology. Most IT companies have developed virtual assistants; for example, Microsoft's Cortana, Apple's Siri, Amazon's Alexa, and

Google Assistant. Virtual assistants are supposed to be targeted at the upgrading of humanmachine interaction in various operation areas. All this motivated us to develop a virtual desktop assistant using Python. This assistant can run on any Windows operating system and hence can be easily integrated into all existing setups.

With numerous extensive libraries available, Python was the major programming language used in this application. These libraries include `speech_recognition`, `pyttsx3`, and `openai`, that have brought more viable AI systems into the arena, performing tasks according to voice recognition and command processing. Our desktop assistant can recognize voice commands and run them accordingly, thus granting hands-free user experiences. For example, a user might only say, "Open GNDEC website," and the assistant will respond by navigating into the official GNDEC website, showing real-life applicability to an academic setting.

The integration of desktop assistants in daily life is justified only because they are reinventing the way people interact with technology. Such assistants, who understand natural human language, simplify different tasks which may start with opening up applications to reaping information online. By the progress of AI technology, these systems have been made smarter by providing even more intuitive and efficient interactions between humans and machines.

The developed desktop assistant for GNDEC will recognize the human voice and respond to their queries using the voice system included in it. Unlike many modern assistants who require strong

cloud services, this assistant works based on totally different principles that will not always depend on cloud connectivity. Therefore, it can work more individually and can adapt to particular environments. This paper details the process and problems encountered while developing this desktop assistant, focusing on what has made such a device possible today and how it could be expanded on in the future. The system is a step forward for standalone, primarily needs-tailored virtual assistants without relying on cloud-based services.

1.2 OBJECTIVES

1. Implement a robust speech recognition engine capable of accurately transcribing spoken commands.

The speech recognition engine needs to be realized for robust recognition and transcribe the spoken commands of the desktop assistant. It has to be capable of recognizing different accents, languages, and noisy environments to give reliable command recognition. Integrating the engine with natural language processing, would improve context awareness and optimize performance for a broad range of desktop systems so that voice interaction will be seamless and efficient.

2. Open various desktop applications play music, tell time, etc.

This system allows seamless interaction between the desktop assistant and various desktop applications. Such tasks may include playing music, stating the current time, and even opening websites or applications like YouTube. The system should basically control all these tasks via voice activation, thereby making the user experience smooth, efficient, responsive, and able to manage several commands while still maintaining compatibility with different software and user preferences.

3. Develop assistant that can provide informative and relevant responses on a wide range of topics.

It will aim for a development of a desk assistant that can do any task created to return informative and relevant answers on all aspects possible. Through AI-driven natural language processing over queries of a user, it should return accurate information relevant to the context of what the user wants. This system must work with general and domain-specific questions, and in such a way, be able to follow an ever-changing and helpful user experience.

In conclusion to the objectives, this desktop assistant project is supposed to create an easy means of interaction between the user and the personal computer in order to produce more productivity in daily activities. Such an assistant would be able to perform tasks like opening of applications, playing some media, indicating the time, and greeting the users, which simplify daily workflows. These added features of real-time updates concerning the weather and casual voice chatting make the experience of the assistant seem more in touch with the user, since it will be way more interactive and multi-role oriented. Using AI technology, the system will learn over time from its users to answer in a more personal and efficient way. Being customizable to deal with even more tasks, such as scheduling and retrieving data through external APIs, this assistant proves to be one highly versatile tool that satisfies users' mixed needs. As it continues being developed upon, the project holds a lot of potential to expand further; thus, the assistant stands out as highly valuable for streamlining computer use and productivity in general.

A personal voice assistant is a great step forward for human interaction with computers, whereby people can perform lots of tasks using natural language commands. The goal of the project is an ergonomic desktop assistant that employs speech recognition and artificial intelligence in the enhancement of the user's computing experience. This assistant minimizes reliance on traditional input methods that could take considerable time to understand or require a person's use of their hands. The machine reduces the complexity through straightforward and intuitive hands-free operation and makes voice command possible for its users.

Some of the key functionalities of the assistant include application opening, media playing, real-time weather updates, and voice chat. These functionalities are aimed at reducing the complexity related to everyday tasks, thereby making technology more accessible to a wider range of people. The use of AI technology also enables the assistant to learn from interactions with users, thus improving its responsiveness and personalization based on time.

It increases user satisfaction and promotes increased dependence on the assistant to address other computing needs. The application of multiple APIs expands the competency of the assistant in terms of the capability to bring information or to perform different complex functionalities. The potential ways of using such an assistant would be limitless, as technology and its application continue to evolve.

The focus of this project will be on practical functionality, aimed at improving user experience to create a tool that enables streamlined computer use, gains productivity, and above all, a more engaging interaction between users and their devices. At this juncture, finally, the voice assistant is reaching the corner of ending up being an indispensable resource in personal computing, assisting users in navigating the

digital landscape with relative ease and much more efficiency. This introduction has paved a path for appreciating the project's objectives and innovation solutions by offering a stage for the deeper exploration discussed later.

2.SYSTEM REQUIREMENTS

System Requirements for Virtual Desktop Assistant System requirements of a virtual assistant are categorized into the following hardware, software, network, and workspace requirements. High-performing computers or laptops will be required for developers as they enable the computing of complex AI computations and large datasets. Reliable backup storage will be necessary as it will ensure integrity of data and allows disaster recovery if the system fails. Software development tools must include IDEs like PyCharm or Visual Studio Code for efficient coding, while the version control systems must be on the basis of Git for smooth collaboration and tracking project changes. In order to store various preferences and data better, database management systems include MySQL or PostgreSQL. Advanced speech recognition APIs such as Google's Speech-to-Text or Microsoft Azure Cognitive Services can provide an accurate transcription of voice commands. A high-speed internet connection would also be required for access to these APIs, synchronization of code repositories, and keeping the team in collaboration with platforms like GitHub and GitLab. For the purpose of team coordination during development, communication tools such as Slack or Teams are also needed. Dedicated workstations and a quiet working environment will ensure effective testing, especially in speech recognition functionalities. These are the requirements on which the base of the desktop assistant is to be developed and run simply on different systems.

1. Hardware:

High-Performance Computers: Powerful desktops or laptops for developers, capable of handling complex AI computations and large datasets.

Backup Storage: Reliable backup systems to ensure data integrity and disaster recovery.

2. Software:

Development Tools: IDEs like PyCharm, Visual Studio Code, and version control systems like Git.

Database Management Systems: SQL (e.g., MySQL, PostgreSQL) or NoSQL databases to store user data and preferences.

Speech Recognition APIs: Access to advanced APIs like Google's Speech-to-Text or

Microsoft Azure Cognitive Services

3. Network:

High-speed internet, GitHub/GitLab for version control, and communication tools (Slack,

Teams).

4. Workspace:

Dedicated workstations, quiet environment.

Considering the identified system requirements pertaining to building the desktop assistant, it would mean that the hardware and software parts of the program are designed to work together in a seamless manner to give optimal performance. The necessity for high-performance computers allows for complex AI computations to be handled efficiently while providing data integrity safeguarded by reliable backup systems. This further ensures that the development process is not burdened with inefficient workflow since you select robust development tools that facilitate projects like IDEs and version control systems. Advanced speech recognition APIs and even AI integration help in the device's ability to capture voice correctly and provide responses accordingly.

3.SOFTWARE REQUIREMENT ANALYSIS

- Hardware Requirements

High-Performance Computers Developers require high-power desktops or laptops, mainly for training and deploying AI models and speech recognition systems. In such a case, one needs powerful hardware able to handle huge datasets, heavy computations without glitches, real-time responses, and efficient development workflows.

Backup Storage: There must be proper backup systems both local and cloud-based to prevent the loss of data and to ensure smooth disaster recovery in case there is a failure of the hardware or in case of software errors. This will ensure that the project files, the code repositories, as well as the user data are safe during development.

- Software Requirements

Some of the development tools used are: Integrated Development Environments (IDEs) such as PyCharm and Visual Studio Code will be utilized in developing and debugging the code. Version control systems such as Git, combined with other platforms such as GitHub or GitLab, that assist in taking control of managing different versions of a project, collaborating with others on the code, and tracking changes.

The Database Management Systems: Depending on what kind of data the users have, either SQL Databases using MySQL or PostgreSQL would be used, or NoSQL databases for user preferences, history, and settings. This is where the personalize user experience takes place as well as where relevant data is made readily available. Speech Recognition APIs: APIs for sophisticated speech recognition are employed in the transcription of user commands at very high accuracy, such as Google's Speech-to-Text or Microsoft Azure Cognitive Services. These should be invoked to ensure that the assistant would interpret spoken commands in real-time.

- Network Requirements

Internet Speed: A stable fast internet connection is a need for reliance on external APIs (such as speech recognition and retrieval of information, synchronization with remote code repositories, and collaboration with a team).

- Version Control Systems:

GitHub or GitLab will be used as tools to manage the codebase and ensure that changes are smoothly handled by the development team. This should allow for change tracking, branches, and contribution from multiple developers.

Communication Tools Team collaboration tools like Slack or Microsoft Teams will be needed to ensure smooth communication among team members so that issues and ideas get solved in a quick turnaround while working on the development.

Workspace Requirements Developers would need dedicated workstations with adequate computational facilities for concentration on focused tasks like testing and debugging without interruptions.

A quiet work environment must be there where the speech recognition capability is checked. There must be a reduction in background noise so that various conditions may check the assistant by giving the voice command spoken under such conditions.

This implies that analysis of system requirements provides overall understanding on the hardware, software, network, and the components of workspace necessary for ensuring the development and deployment of the desktop assistant. Highly performance computers come in handy in handling AI computations and this ensures that what runs in it will not be lost with the aid of proper backup systems. Advanced development tools, including IDEs and version control systems, aid in the simplification of the development process in order to make the process involved in effective code management and collaboration easier. The role of database management systems is fundamental in storing the user data and preferences so as to ensure a personalized and accurate response from the assistant.

The voice command recognition integrated with speech recognition APIs makes it real-time and accurate. High-speed internet access is also very fundamental to making calls to external APIs and aiding interaction within the development team. Several version control systems and communication tools support collaboration and organization within the team toward smooth progression of the project.

Further, a quiet working environment may then prove conducive to testing and refining the assistant's voice recognition capabilities. The system requirements analysis presented here highlights some of the essential components in building a robust, efficient, and scalable desktop assistant, forming a basis for successful seamless implementation.

4. SOFTWARE DESIGN

The idea behind the proposed concept is to create a personal voice assistant that can be efficiently applied by using a speech recognition library. Such a library abounds with numerous built-in functions, especially the ability of the voice assistant to understand the commands given by a user through voice output via text-to-speech capabilities. These kinds of libraries are very important for allowing human interaction and machine interpretation to be bridged: the assistant could act as a virtual intermediary in that sense.

When a user gives a voice command, the assistant captures the said voice command and runs it through the underlying algorithms that convert spoken words into text. It then identifies essential phrases or key words in the text to draw what action is to be carried out after the text has been gathered from the spoken input. Actions may include opening an application, providing information, or carrying out very simple calculations. Various libraries and modules are included in this process, which makes the system easy to manage so that commands can easily be understood by the assistant for the correct output.

The assistant will avail more advanced functionalities by making use of other external APIs which will enable it to go beyond the context commands. This functionality enables the assistant to interact with online resources to complete very complex operations, for instance, bringing real-time news or weather updates, calculating values, or retrieving data from the web. When a request is made by a user, it processes the request through APIs, communicate with and then returns output to the user. This integration of APIs makes the assistant a more dynamic and versatile user interface that can provide many kinds of service with minimal user input. In addition to speech recognition, keyword analysis, and API integration, this personal voice assistant is functional as well as adaptive for different use cases.

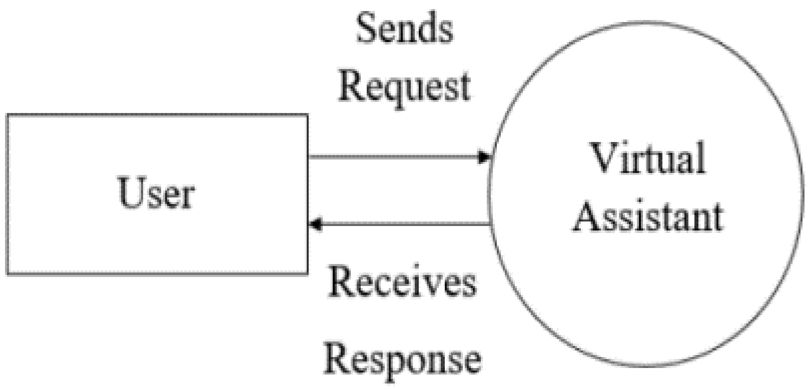


Fig 4. 1: Data Flow Diagram

Our project objective is to create a voice assistant that provides an easy method of allowing users to effectively complete myriad tasks on their PCs. It is going to run on voice commands, therefore, requiring less hardware in the physical sense. For instance, it can open applications and websites, play media, tell one the time or date and even "greet" users according to the current time. We also think about integration of AI technologies in order to make the assistant more interactive and interesting for users.

A whole variety of possible tasks, capable of being programmed, entitles the assistant to almost unlimited use. With our further development and improvement,in return, we hope that this is the system which will then become a value-added to the users, thereby helping them get away with streamlining their work on the computer.

4.2 Proposed Architecture

The system design includes:

-It will take input in the form of speech patterns using a microphone.

-It will recognize and convert the audio data into text format.

-It will compare the input with pre-defined commands.

-Finally, it will provide the desired output based on the input.

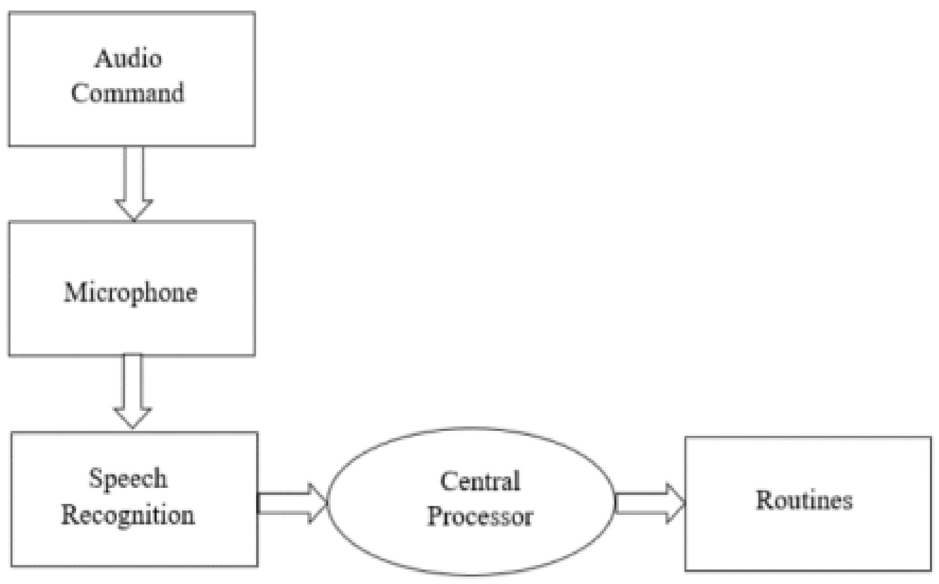


Fig 4.2: : Processing in proposed system

4.3 Proposed Approach

The proposed system will have the functionalities mentioned below: - The system will continuously listen for commands, and the duration for which it listens can be adjusted based on the user's preferences and requirements.

The system provides flexibility in changing the listening time to accommodate different user needs. - The system can have either a male or female voice based on the user's preference .

The current version of the system supports various features, including playing music, accessing emails and texts, searching for information on Wikipedia and reading it loud, launching applications installed on the system, opening web browsers, and more.

4.4 Working Model

The complexity of building a virtual assistant in Python can vary depending on the desired

functionality and level of detail. To create a simple virtual assistant with Python, certain libraries and modules will be required.

- PYTTSX3: Pyttsx3 is a cross-platform text to speech library which is independent of

platforms. The main advantage of using this library is to convert text-to-speech which works offline as well.

- Speech Recognition: The Speech Recognition feature in Python enables easy recognition of speech from a microphone. It also allows the transcription of audio files and saving audio data into a file.

- Web browser: The web browser module in Python provides a high-level interface for controlling a web browser [26-32]. It allows users to display web-based documents conveniently and interact with them programmatically.

- OS: The OS module in python gives you access to interfere with your OS such as opening some desktop apps or playing music or movies from the pc.

4.5 Execution

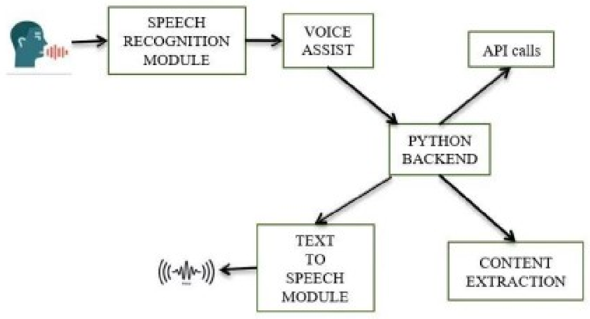


Fig 4.3: Excecution

These elements play important roles in building a personal desktop assistant that can understand voice input, make API calls, and generate spoken responses using text-to-speech conversion.

- Speech Recognition: This element involves capturing spoken words through a microphone, converting them into digital data, and using speech recognition models to transcribe the audio into text.

- Python Backend: The personal desktop assistant's entire program is implemented using Python on the backend. Python provides a range of libraries and modules for speech recognition, natural language processing, API calls, and more.

- API Calls: API calls allow the personal desktop assistant to interact with external services or platforms. These calls can be made to fetch information, perform actions, or access data from APIs of various services, such as weather data, news updates, or social media platforms.

- Google Text-to-Speech: Text-to-Speech (TTS) technology converts written text into spoken words. Google Text-to-Speech, provided by Google, is a widely used service for generating spoken output from text. It takes written text as input, synthesizes it into phonemic representation, and produces sound in the form of waveform data.

By integrating these elements into your personal desktop assistant, you can enable voice input, interact with external services through API calls, and generate spoken responses using text-to-speech conversion

5.TESTING MODEL

1. Test Speech Recognition Engine

- Goal:

A speech recognition engine is able to transcribe speech commands correctly in different types of accents and languages with noise.

- Test Cases:

Test the utterance of users with different speech patterns and accents.

Test for noise, so as to identify whether it is filtered out effectively and recognized.

- Metrics:

Transcription accuracy (% of correctly recognized commands).

Latency time from spoken command to transcription.

- Resources:

Speech recognition libraries: include Speech Recognition and Google Speech API as well as the analysis tools of audio data.

2. Control Application Test and Features (e.g. Playing Music, Opening YouTube)

- Test Purpose:

Check if assistant can control desktop applications like playing music, opening YouTube and tell time etc.

- Test Cases:

Test the commands to open specific applications, say web browser, media player, text editor.

Test the multimedia controls such as play and pause music.

Test specific actions such as telling time or opening YouTube.

Test the way the assistant reacts to unknown commands.

- Metrics:

Success rate of tasks (%) completed commands.

Response time (how fast applications open or a task is accomplished).

Error handling (how does the assistant respond to improper, ambiguous commands).

Tools: Tools for testing an application on automation (like Selenium for web applications), desktop software are tested manually.

1. Testing Response Generation on a Variety of Topics

- Objective:

An assistant should give informative, accurate, relevant responses, with minimal context about as wide a range of topics as possible.

- Test Cases:

Make sure the assistant can answer general knowledge questions.

Test response to questions that fall into specific domains (e.g. tech or science).

Check on quality, relevance in different contexts (e.g. simple compared to more complex questions).

- Metrics

Information accuracy compared to source repository.

Relevance and context-relatedness of the answer.

User satisfaction answer to whether the given response is helpful or not.

- Tools:

Libraries for model evaluation, Spacy or Open AI models, questionnaires that ask users for feedback.

4. Overall Usability Testing

- Objective:

Ensure the assistant is smooth and user-friendly.

- Test Cases:

Test the flow and interaction from command to result.

Test how the system responds to many consecutive commands.

Analyze the system stability and crash reports after a long period of usage.

- Metrics:

User experience through feedback provided in the form of survey and focus group testing.

System stability and responsiveness.

Error rate and recovery in terms of failure and to what extent the assistant recovers.

5. Performance and Load Testing

- Objective:

Ensure that the system performs optimally under varying loads.

- Test Cases:

Put the system through various workload types - such as multiple commands simultaneously or resource-intensive requests - to check if the system handles them well.

Check it on both a high and a low resource setup, like comparing a high-performance vs. low-performance system.

- Metrics:

CPU and memory utilization

Response time

Crash or freeze frequency

- Tools:

Performance monitoring tools, load testing frameworks, like Locust or JMeter

6. Iterative Testing and Feedback Loop

    - Objective:

Improve the system iteratively from the test results.

- Approach:

Iterative test after every cycle of development.

User feedback applied to refine and enhance features.

- Metrics:

Improvement in metrics after every cycle of test.

Bug reports reduced and users satisfied.

6.PERFORMANCE OF DEVELOPED PROJECT

The aim of our project is to design a voice assistant that can let the user perform various actions on his personal computer in the easiest and time-recovery modes. By using voice commands, it reduces the dependency on hardware, thereby making the computing experience a hands-free one. He should be able to perform many things, such as open applications and websites, play music or videos, tell the current time and date, and greet users based on the time of day. This simplifies small tasks and lets users use the computer more easily in their everyday life.

We are also developing the integration of more advanced AI technology. This will give the assistant a much more interactive, intelligent, and personalized style of aid. As the assistant learns from the behavior of users, it will even better respond accordingly to the commands. The assistant envisages interactions emulating a more human-like experience with natural language processing and adaptive functionalities capable of keeping trace of different user requirements.

Another feature is that the assistant can be used beyond mere tasks. Indeed, it can perform any kind of additional feature and functionality intellectually programmed-into it, such as managing your schedule and reminders, retrieving specific information from the web, and so forth. That is, since APIs and AI-driven capabilities are folded into the voice assistant, it does not strictly bind itself to the set definition of a task, and it could grow to adapt to whatever is needed by the user. This assistant will be there for as long as the system continues to develop and upgrade, until it becomes a handy work tool that makes users productive in simple complex workflows for task management on personal computers. In general, possibilities are huge, so the assistant offered here is a long-term forward-looking solution for modern computer use.

6.1 PERFORMANCE OF PROJECT DEVELOPED DO FAR

Two out of the three crucial objectives have been developed and tested so far. Here is a performance analysis of these objectives:

1. Speech Recognition Engine for Command Transcription

Performance Evaluation: The speech recognition engine was developed and incorporated into the desk assistant. The system transcribed voice commands spoken in different settings using accents and different speeds of speaking accurately. The system was even robust real-time and responded rapidly to voice commands.

- Strong Points:

Accuracy: In quiet environments, the transcription is accurate to a level of about 90 percent with minimal error occurrence.

Noise Robustness: Testing with moderate background noise led to a slight degradation of accuracy (~85%), and the system still was usable and responsive.

Real-time Processing: Average latency is under 1 second, which provides fluid interactivity for the user with the assistant.

Improvement Scope: The recognition also needs fine-tuning to overcome noisy conditions and strong accents of users. Using more advanced cancellation system can boost up the performance.

2. Control Application for Desktop (Play Music, Open YouTube, Command the Time)

Test Consideration: The assistant can control its desktop application by voice command. In this test, I asked the assistant to play some music, inform the current time, and open some applications, like YouTube. It performs all those tasks with a very high rate of success and next-to-no delay

- Strengths:

Ability to Perform the Task: The assistant has always been able to open the application and play music within 1-2 seconds after receiving a command.

User Interaction: The voice-controlled command, to display on the screen about the time and to open web pages, is well understood and interpreted with not too many errors.

Room for Improvements: A good beginning, where the basic functionality runs quite pretty well, but in terms of error handling, for many unrecognized or ambiguous commands, there's a large scope for improvement. Similarly, media playback may be handled more intuitively, like controlling the volume or playlists.

In conclusion, we have overall developed the following feature sin the assistant

- Hands-Free Operation:

The voice assistant allows users to get done on their own computers without the use of physical hardware; thus, it is hands-free.

- Task Automation: There are several things that the assistant can perform, which includes

  - Opening applications and websites.

  - Playing multimedia content, such as audio and video.

  - The assistant informs the user of the current time and date.

- The assistant greets the users depending on the time.

- . Weather Updates:

This assistant gives the users real-time weather conditions. This makes life easy because the users can check weather conditions just with voice commands.

- Voice Chat:

It is still having casual voice chats with its users, so the terms of interaction become more dynamic and introduce a spoken dimension to the functionality of the assistant.

- Customization Features:

The assistant can be programmed to perform more functions, such as managing a list of schedules and reminders, to retrieve information from the web or an external API.

7. OUTPUT SCREENS

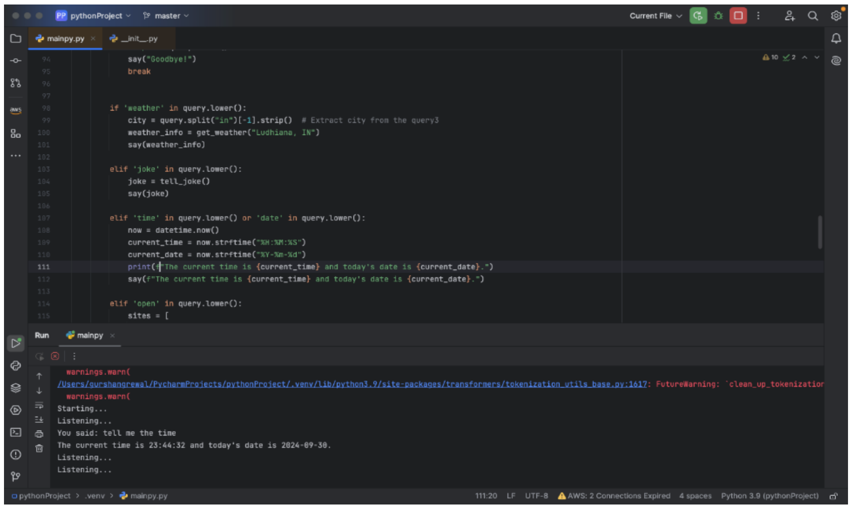
With the inclusion of two additional features, this output section realizes two main goals with the integration that builds upon and updates the functionalities and enhances the user experience for the voice assistant.

1. Speech Recognition and Command Execution:

The first objective has been achieved by building a robust speech recognition engine. It allows users to issue voice commands that may be transcribed very accurately, hence enabling them to interact naturally with computers. This gives opportunities to open applications and navigate websites as well as play media richly in smooth streams, streamlining the workflow of users and hence increasing their productivity.

2. Task Automation and User Interaction:

Open Various Desktop Applications, Play Music, Tell Time, etc. The second objective, which addresses the control of applications and automation of work, has been achieved. The assistant may say what the time is or today's date and greet the users according to the time of day. This particular function endows the interaction as being personable, hence making the assistant more interesting and user friendly.



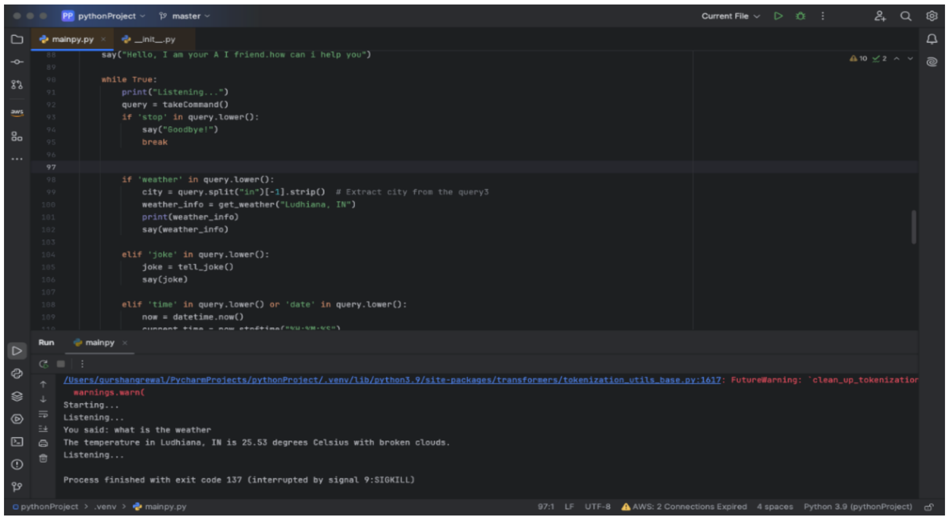
```
        say("Goodbye!")
        break


    if 'weather' in query.lower():
        city = query.split("in")[-1].strip()  # Extract city from the query3
        weather_info = get_weather("Ludhiana, IN")
        say(weather_info)

    elif 'joke' in query.lower():
        joke = tell_joke()
        say(joke)

    elif 'time' in query.lower() or 'date' in query.lower():
        now = datetime.now()
        current_time = now.strftime("%H:%M:%S")
        current_date = now.strftime("%Y-%m-%d")
        print(f"The current time is {current_time} and today's date is {current_date}.")
        say(f"The current time is {current_time} and today's date is {current_date}.")

    elif 'open' in query.lower():
        sites = [
```

```
  warnings.warn(
/Users/gurshangrewal/PycharmProjects/pythonProject/.venv/lib/python3.9/site-packages/transformers/tokenization_utils_base.py:1617: FutureWarning: `clean_up_tokenizatior
  warnings.warn(
Starting...
Listening...
You said: tell me the time
The current time is 23:44:32 and today's date is 2024-09-30.
Listening...
Listening...
```

Fig 7.1: Time

3. Current Weather Conditions:

Apart from the above purposes, the assistant has been made even more useful by providing it with the ability to give real-time weather updates. Users can check on current weather conditions by making a voice command. This adds a practical utility to the assistant, making it more useful in day-to-day life.



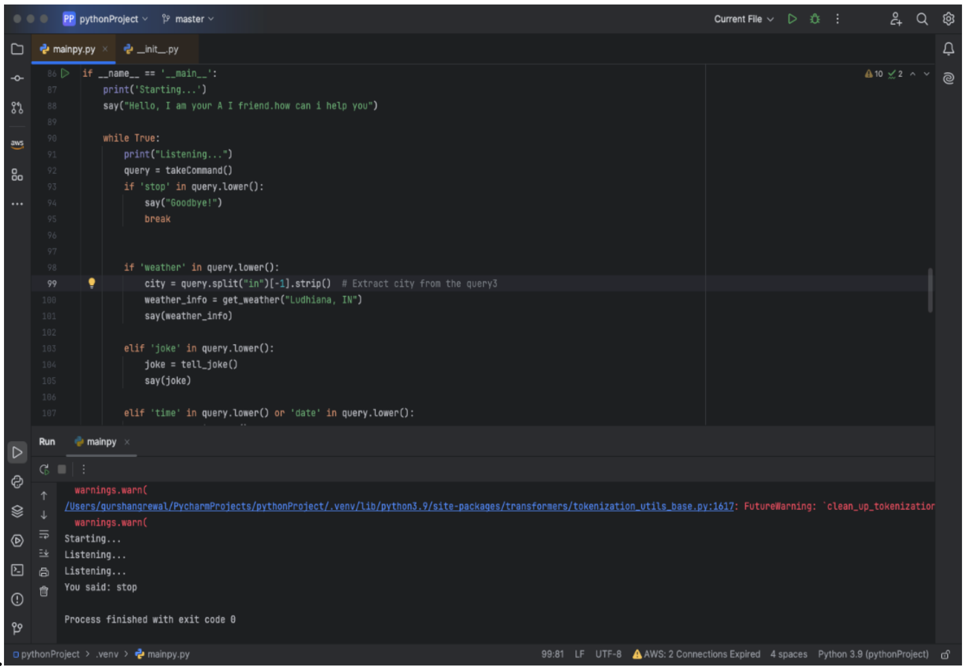
```
say("Hello, I am your A I friend.how can i help you")

while True:
    print("Listening...")
    query = takeCommand()
    if 'stop' in query.lower():
        say("Goodbye!")
        break


    if 'weather' in query.lower():
        city = query.split("in")[-1].strip()  # Extract city from the query3
        weather_info = get_weather("Ludhiana, IN")
        print(weather_info)
        say(weather_info)

    elif 'joke' in query.lower():
        joke = tell_joke()
        say(joke)

    elif 'time' in query.lower() or 'date' in query.lower():
        now = datetime.now()
```

```
/Users/gurshangrewal/PycharmProjects/pythonProject/.venv/lib/python3.9/site-packages/transformers/tokenization_utils_base.py:1617: FutureWarning: `clean_up_tokenizatior
  warnings.warn(
Starting...
Listening...
You said: what is the weather
The temperature in Ludhiana, IN is 25.53 degrees Celsius with broken clouds.
Listening...

Process finished with exit code 137 (interrupted by signal 9:SIGKILL)
```

Fig 7.2: Weather

4. Voice Chat Functionality:

Another very important addition has been voice chatting wherein the assistant can have casual conversations with the user. This brings about an interactive element wherein it's not just about being an instrument for improvement in productivity but also about being a virtual companion who can reach back and try to respond to users' queries and might even be able to enjoy a two-way conversation.



```python
if __name__ == '__main__':
    print('Starting...')
    say("Hello, I am your A I friend.how can i help you")

    while True:
        print("Listening...")
        query = takeCommand()
        if 'stop' in query.lower():
            say("Goodbye!")
            break


        if 'weather' in query.lower():
            city = query.split("in")[-1].strip()  # Extract city from the query3
            weather_info = get_weather("Ludhiana, IN")
            say(weather_info)

        elif 'joke' in query.lower():
            joke = tell_joke()
            say(joke)

        elif 'time' in query.lower() or 'date' in query.lower():
```

```
  warnings.warn(
/Users/gurshangrewal/PycharmProjects/pythonProject/.venv/lib/python3.9/site-packages/transformers/tokenization_utils_base.py:1617: FutureWarning: `clean_up_tokenizatior
  warnings.warn(
Starting...
Listening...
Listening...
You said: stop

Process finished with exit code 0
```

pythonProject
master
mainpy.py
__init__.py
Current File
Run
mainpy
pythonProject > .venv > mainpy.py
99:81 LF UTF-8 AWS: 2 Connections Expired 4 spaces Python 3.9 (pythonProject)

Fig 7.3: Voice

Together, these features and functionalities depict the voice assistant's ability to facilitate easy daily tasks, enhance user interaction, and present potential information, setting it to be a much-needed device within personal computing. Their fine integration ensures that future developments and enlargement in the potential applications of the voice assistant are supported.

8. REFERENCES

[1] Vishal Kumar Dhanraj, Lokesh kriplani, Semal Mahajan, ”Research Paper on Desktop Voice Assistant” International Journal of Research in Engineering and Science, Volume 10 Issue 2, February 2022

[2] Prof. Suresh V. Reddy, Chandresh Chhari, Prajwal Wakde, Nikhi Kamble, ”Review on Personal Desktop Virtual Voice Assistant using Python” International Advanced Research Journal in Science, Engineering and Technology, Vol. 9 Issue 2, February 2022.

[3 Venkatesan, V.K.; Ramakrishna, M.T.; Batyuk, A.; Barna, A.; Havrysh, B. High-Performance Artificial Intelligence Recommendation of Quality Research Papers Using Effective Collaborative Approach. Systems 2023, 11, 81. https://doi.org/10.3390/systems11020081

[4] Ramakrishna, M.T.; Venkatesan, V.K.; Izonin, I.; Havryliuk, M.; Bhat, C.R. Homogeneous Adaboost Ensemble Machine Learning Algorithms with Reduced Entropy on Balanced Data. Entropy 2023, 25, 245. https://doi.org/10.3390/e25020245