INTELLINEO - An Intelligent PersonalAssistant

[1]Muktanandh Bombothu
Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
muktanandhbombothu@gmail.com

[2]Yaseen Abdul
Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
abdulyaseen662@gmail.com

[3]Umasri Katragadda
Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
Umasrikatragadda2002@gmail.com

[4]DS Bhupal Naik
Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
dsbhupal@gmail.com

***Abstract*—Man-made consciousness means to lessen our physical association with electronic devices. Using voice, body language, and other techniques, this technology is currently being developed to enhance our natural interactions with gadgets. Because they enable us to carry out our day-to-day responsibilities with straightforward commands, virtual personal assistants (VPAs) are rapidly gaining popularity. Virtual personal assistants (VPAs) based on various use cases have been developed by numerous multinational corporations, including intelligent devices like Echodot and Alexa. These virtual assistants are having a drawback of security and data theft, So here IntelliNeo, The intelligent personal assistant is designed in such a way that it addresses the security aspects and also allow us to customize our Virtual Personal Assistant (VPA) based on our needs. The methodology of building IntelliNeo involves Voice Acknowledgment, Text Processing, and the added advantage of active background noise cancellation makes our assistant stand out from the competition. Ultimately the purpose of IntelliNeo is to effortlessly automate the daily tasks of its user. *Index Terms*—pyttsx3, PyQt5 tools, Mail Automation, WhatsApp Automation, Text processing, pyautogui, Google Speech Recognition, Microsoft Sapi5.**

Keywords—pyttsx3, PyQt5 tools, Mail Automation, WhatsApp Automation,Text processing, pyautogui, Google Speech Recognition, Microsoft Sapi5.

I. INTRODUCTION

The motto of our study is to develop an application that can serve as our personal assistant. Time is the most precious thing in the world, We should not waste that, and everyone want them to be safe, We mostly prefer to reduce our physical involvements in doing things in our present world, this assistant is mostly designed and developed for us, we call it using the term 'Automation' and we are running after that in our present world. Our IntelliNeo is almost similar to Siri, the first Virtual Personal Assistant, made its debut in 2012 but was not made by Apple. As the Artificial intelligence became popular and they started implementing in everywhere, it was also implemented in virtual assistants, which are used for understanding and doing the tasks assigned by the humans We can convert from voice into text and text to voice using python.

979-8-3503-1494-6/23/$31.00 ©2023 IEEE

A Python library also called package that is known as the speech recognition API enables developers to incorporate speech recognition functionality into their Python applications. It gives a method for changing over human discourse into text design, which can be utilized for many purposes, for example, voice-controlled interfaces, robotized records, and text processing. One of the most famous speech acknowledgment APIs in Python is the Speech Recognition library, which gives a straightforward to-utilize connection point to perceive dis- course from different sources, for example, mouthpieces, sound documents, and pre-recorded sound streams. Just by saying the commands orally we can make computers work and do our assigned tasks. The user's voice must be clear and pronounced correctly to get recognized properly by the speech recognition Api provided by Google. This issue always exists to some extent as one cannot eliminate the noise. The following are a few unique functionalities of IntelliNeo.

1) Interactive GUI
2) Face Authenticated start and wakeup
3) News scrapper
4) game playing
5) Image scrapper
6) Nasa information scrapper
7) WhatsApp automation
8) Active background noise cancellation

Fig. 1. INTELLINEO logo

2023 International Conference on Quantum Technologies, Communications, Computing, Hardware and Embedded Systems Security (iQ-CCHESS) | 979-8-3503-1494-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/iQ-CCHESS56596.2023.10391450

A quick and efficient form of communication is voice communication. Many people prefer speaking to writing. Our IntelliNeo application accepts audio clips for voice input signals. Alexa, Google Assistant, and Siri use some algorithms and servers to get- ting better precision, quickness, and contextual awareness. The main difference between them is how they all introduce data privacy difficulties and different methods. For the virtual assistant, internet access is important. In response to a user's command, the request. Our primary goal is to develop a personal assistant which is easy and safe to interact with.

II. LITERATURE SURVEY

The application created in [1] act as a Virtual Personal Assistant and they have added speech recognition in their application, and they also added features like weather monitoring, and more features like sending emails to a particular person and browsing through the internet.

A study on previously present virtual assistants and they have created a new virtual assistant named "*FRIDAY*", it does a lot of tasks like playing audio and video, weather monitoring and browsing through the internet, booking a taxi and ordering some food and doing some scientific calculations and a lot more features [2].

Designed a virtual assistant named "*ZIRA*", they used a lot of voice recognition modules in this and also added many features to their virtual assistant [3].

The removal of noise from the audio files that are taken as input files [4]. The noise cancellation was done by using various machine learning algorithms and other modules to retrieve the data from the files.

The virtual assistant "jarvis" is made possible using python's libraries.

An endeavor is made here to build a wise voice individual aide utilizing Python, which gives the ability to work gadgets given voice or discourse [6].

chatbots can replace humans in tedious positions of noting inquiries and giving productive reactions and many other redundant tasks. [7].

P.Krishnaraj et al. presented a project on Portable Voice Recognition which was funded by the National Science Foundation. The precision of this Text to Speech (TTS) system engine is rather small and lacks IoT [19].

The frequent issues here are there is no security here, that anyone with this program can run this, there is no face recognition system in their application, but we added the face recognition system into our application until you are the authorized person you cannot go ahead or give commands or make our virtual personal assistant do some tasks, and there is no background noise cancellation here, but we added background noise cancellation here, another major issue here is there is no interface or UI for the personal assistant every time we have to run that program and it doesn't have a GUI, and it's not a complete application, but in our project, we added GUI to our application using PyQt5 tools and We connected the front end and the background, thereby creating a complete application and we also added some extra features like WhatsApp messaging and automation of YouTube and the automation of the google chrome and we also added a game, which is created by our own by using pygame, which are a set of modules used for game designing in python and we also added Image scraper and NASA news extractor, etc,...

III. METHODOLOGY

A. Problem Statement

Developing a virtual personal assistant that can send emails without the use of a keyboard to any specific person mentioned by the user, and for setting a reminder for a particular task, and for playing music we can also integrate it with Spotify or music-related applications and for watching movies, telling the weather report of our surroundings or a particular place, providing effective search through the internet like google , Wikipedia, and for sending the WhatsApp messages to our friend or even calling them using WhatsApp, and for the automation of YouTube, and the automation of google chrome, and also for providing location services and for playing games, etc.,

B. The Proposed Method

- Our target is to develop an intelligent personal assistant which has the capability of automating everyday tasks.
- We will be requiring a set of libraries that can provide the functionalities of text processing and analytics as well as some APIs.
- Designing functions for each operation, So that the appropriate action will be carried out.
- After completing the code, the voice assistant continuously listens for user input until the user exits with his question because that is a variable that sets the audio processing time change and adapts according to the user's needs.
- If it is assumed that IntelliNeo was not able to hear you, It requests you to repeat it until the voice is correctly identified.
- Finally, if the user enables the voice assistant can find it new, ie. Gmail automation, Newsreader, playing games, downloading YouTube videos, and many more.

C. System Architecture

- The project's design/architecture depicts IntelliNeo's flow of execution. It highlights the required software components for the project. It contains the following sequence of operations, each with its own set of requirements.
  The graphic representation of that in figure 1 is shown below. It operates as follows: When the device's microphone senses any incoming voice, our assistant uses this

as a source and analyzes the obtained data using Google's speech recognition API and checks whether there exists any appropriate action to be executed.

- The figure 2 below represents the features and requirements that IntelliNeo is capable to perform and the Python Modules which we have incorporated to achieve the same.
- The figure 3 represents the design/system architecture of IntelliNeo which includes Python, a Local database in the backend. The middleware contains various APIs and modules. The top layer contains some of the functionalities of IntelliNeo.

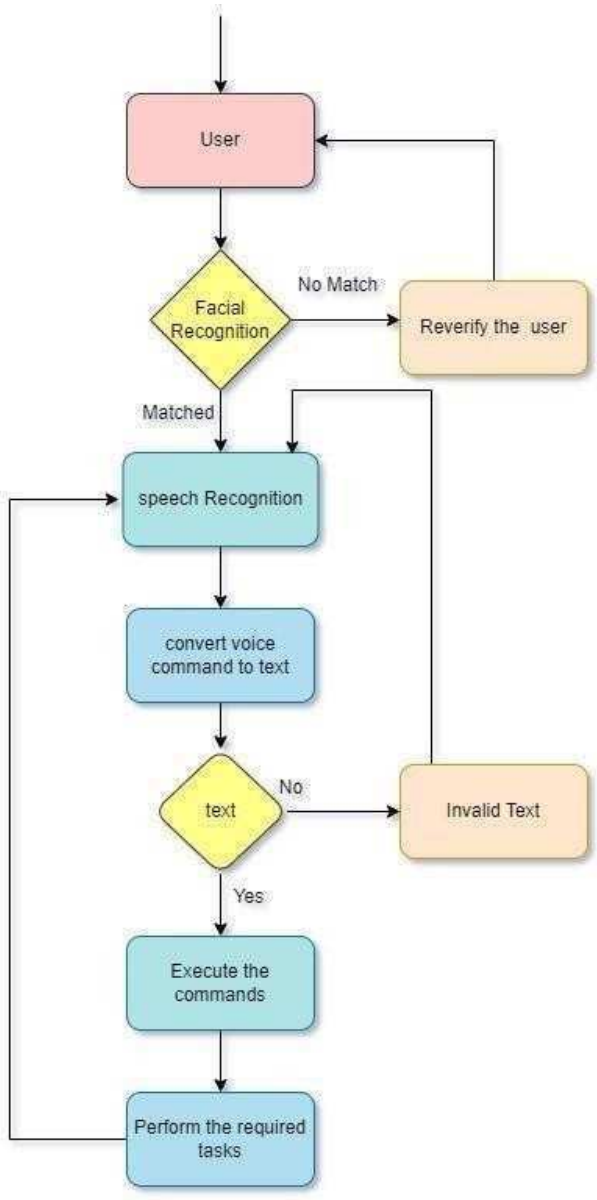

Fig. 2. Proposed architecture of INTELLINEO

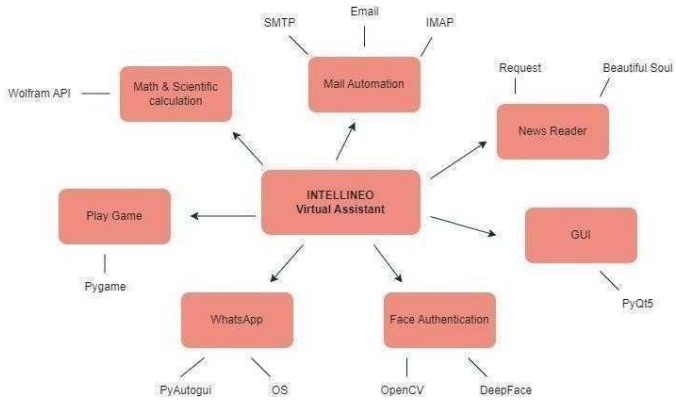

Fig. 3. Features and Requirements of INTELLINEO

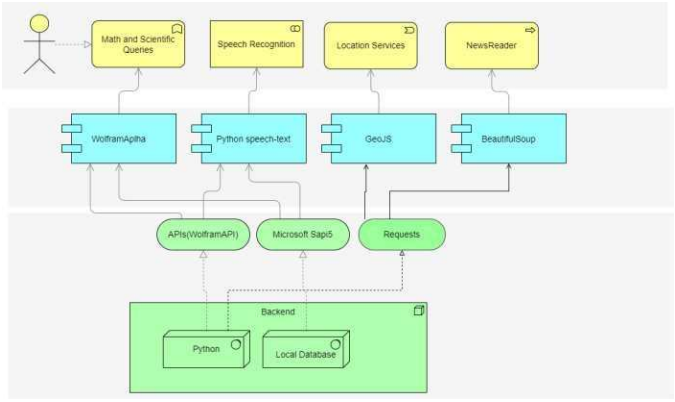

Fig. 4. System Architecture of INTELLINEO

IV. IMPLEMENTATION

A. Calculating distance from your location to any other location on the globe.

The distance that separates two points on the surface of a sphere, like the Earth, is calculated mathematically using the Haversine formula. In applications for navigation and geolocation, the distances between sites are frequently used because routes, trip times, and other information must be calculated.

```
def calculate(lat1, lon1, lat2, lon2):
    R = 6371 # This is the radius of earth in kilometers
    dLat = math.radians(lat2 - lat1)
    dLon = math.radians(lon2 - lon1)
    a = math.sin(dLat/2) * math.sin(dLat/2) + math.cos(math.radians(lat1)) \
        * math.cos(math.radians(lat2)) * math.sin(dLon/2) * math.sin(dLon/2)
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
    distance = R * c
    return distance
```

Fig. 5. Distance calculation using Haversine formulae

The Haversine of an angle is the same as the square of the sine of half the angle. The Haversine formula employs the latitude and longitude of two locations to determine the great-circle distance, which is the shortest distance between two points on a sphere's surface.

Algorithm 1 : Haversine

Input : lat1, long1, lat2, long2
$R \leftarrow 6371$ *radiusofearth*(*in kms*) $dLat \leftarrow$ *math.radians*($lat2 - lat1$) $dLon \leftarrow$ *math.radians*($lon2 - lon1$)
$a \leftarrow$ *math.sin*($dLat/2$) $*$ *math.sin*($dLat/2$) + *math.cos*(*math.radians*($lat1$)) *math.cos*(*math.radians*($lat2$)) $*$ *math.sin*($dLon/2$) $*$ *math.sin*($dLon/2$)
C $\leftarrow 2 *$ *math.atan2*(*math.sqrt*(a), *math.sqrt*($1 - a$))
distance $\leftarrow R * C$
Output : distance in (kms)

B. Automating Email tasks

i. Sending mails

```
def send_mail():
    speak("Whom do you want to send the mail sir.")
    mail=listen()
    newm=""
    for i in range(len(mail)):
        if mail[i] != " ":
            newm+=mail[i]
    newm=newm+"@gmail.com"
    receiver = newm
    sub = "testing email automation "
    speak("what is the message sir.")
    mess = listen()
    body = str(mess)

    em['From'] = sender
    em["To"] = receiver
    em['Subject'] = sub
    em.set_content(body)

    context = ssl.create_default_context()

    with smtplib.SMTP_SSL('smtp.gmail.com',465,context = context) as smtp:
        smtp.login(sender,empass)
        smtp.sendmail(sender,receiver,em.as_string())

    speak("Email sent sir.")
```

Fig. 6. Sending email message using SMTP and Gmail server

This Python function is designed to make it easier for a user to send an email using their Gmail account. Rather than having to navigate to the Gmail website and manually compose and send an email, the user can simply call this function in their Python script and interact with it using voice into text and text into voice technology. The function first prompts the user to speak the email address of the recipient they want to send the email to. It then adds "@gmail.com" to the end of the email address to create the full recipient address.

Algortihm 2 : Sending mails

Input : receiver's mail, message
mail$\leftarrow$ *call listen*()
Newm $\leftarrow$ *emptystring*
while $i \leq$ *length*(*mail*) **do** :
  if mail[i] = "" :
    newm $\leftarrow$ *mail*[*i*]
  end if
end while
newm $\leftarrow$ *newm* + "*@gmail.com*"
receiver $\leftarrow$ *newm*
speak("*what is the message sir.*")
mess $\leftarrow$ *listen*()
body(*mess*)
em[$'From'$] $\leftarrow$ *sender*
em["*To*"] $\leftarrow$ *receiver*
em[$'Subject'$] $\leftarrow$ *sub*
em.set(*body*)
context $\leftarrow$ *ssl.create*()
with smtplib.SMTP-SSL('smtp.gmail.com',465,context = context) **as** smtp:
  smtp $\leftarrow$ *login*(*sender, empass*)
  smtp $\leftarrow$ *sendmail*(*sender, receiver, em*)
  Output : mail sent successfully

The Email Message class, which we are using to create an email message object, specifies the essential characteristics, including the sender, recipient, subject, and body of the email. It then establishes a secure connection to Gmail's SMTP server using the smtplib module, creates an SSL context, logs in using the sender's email address and password, then sends the email message to the recipient .

ii. Read Mail Inbox

```
def read_mail():
    imap_server = "imap.gmail.com"
    import imaplib
    imap = imaplib.IMAP4_SSL(imap_server)
    imap.login(sender,empass)
    imap.select("inbox")
    response, messages=imap.search(None, 'UnSeen')
    speak("I am checking for new mails sir.")
    if len(messages) <=1 and messages[0] == b'':
        print("No new mails")
        speak("No new mails sir")
    else:
        messages = messages[0].split()
        msgcount = len(messages)
        latest = int(messages[-1])
        oldest = int(messages[0])
        for i in range(latest, latest-msgcount, -1):
            # fetching mails from inbox using imap
            res, msg = imap.fetch(str(i), "(RFC822)")
            for response in msg:
                if isinstance(response, tuple):
                    msg = email.message_from_bytes(response[1])
                    print()
                    print(msg["Date"])
                    print(msg["From"])
                    print(msg["Subject"])
                    speak("Mail from "+str(msg["From"]))
                    speak(str(msg["Subject"]))
```

Fig. 7. Receiving mails from inbox using IMAP

readmail() connects to a Google account via IMAP and scans the inbox for any unread emails. If there are any unread emails, it speaks the sender and topic of each email after retrieving the date, sender, subject, and text of the email. The code connects to the IMAP server and parses the email messages using the

email libraries. It says no new mails if all messages are read.

Algorithm 3 : Reading mails

Input : user's mail inbox access using imap

imap − server ← "*imap.gmail.com*"
import imaplib library

imap ← *selectingimapserver*
imap ← *login*(*sender, empass*)
imap ← *select*("*inbox*")
response, messages ← *imap.search*(*None,' UnSeen'*)
call speak("checking for new mails sir.")

if *length* ≤ 1 *and msg*[0] ← *b* **then** :
print("*Nonewmails*")
end if

Output : read aloud inbox mails

C. News Reader

```
request = req.get('https://www.bbc.com/news')
html = request.content
soup = BeautifulSoup(html,'html.parser')

def newsReader():
    nl = []
    for h in soup.find_all('h3',class_='gs-c-promo-heading__title'):
        news_title = h.contents[0].lower()
        if news_title not in nl:
            if 'bbc' not in news_title:
                nl.append(news_title)

    speak("sir, here are some news that i fetched as per your request")
    for i in nl[:6]:
        speak(str(i))
```

Fig. 8. Retrieving news articles from BBC

Beautiful Soup is used to extract titles from the BBC news website. To locate all the news titles on the page, it sends a request to the website and parses the HTML information. Following that, it lowercases the titles and eliminates any that are duplicates or contain the letter "bbc." The code outputs the first 6 news headlines to the user while speaking the news titles.

Algorithm 4 : News Reader

Input : voice command for news

request ← "*req.get*(*'https : //www.bbc.com/news'*)"*html* ← *request.content*
soup ← *BeautifulSoup*(*html,' html.parser'*)
nl = []

for *h* ← *soup.find*(*'h3tags'*) **do** :
news-title ← *convert title lowercase*
if *newstitle* ⊨ *nl* **then**:
if *'bbc'* ⊨ *news − title* **then**:
nl ← (*news − title*)
end if
end if

Output : read aloud news articles from BBC

V. RESULTS

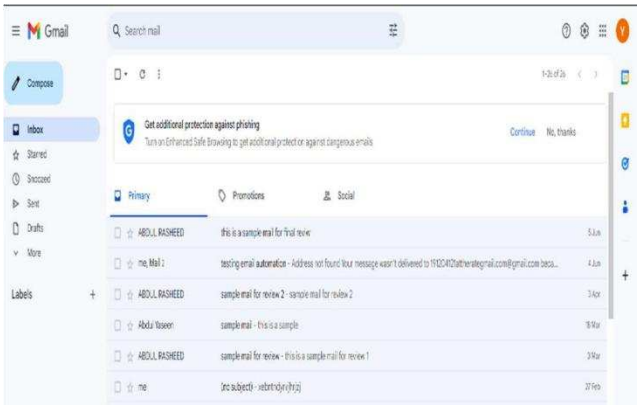

Fig. 8. Mails sent and read by IntelliNeo

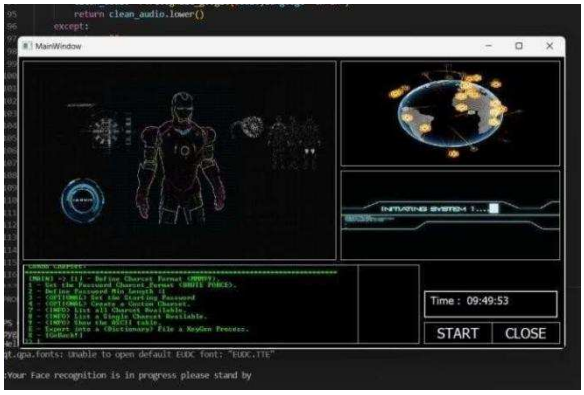

Fig. 9. Interactive gui for our VPA

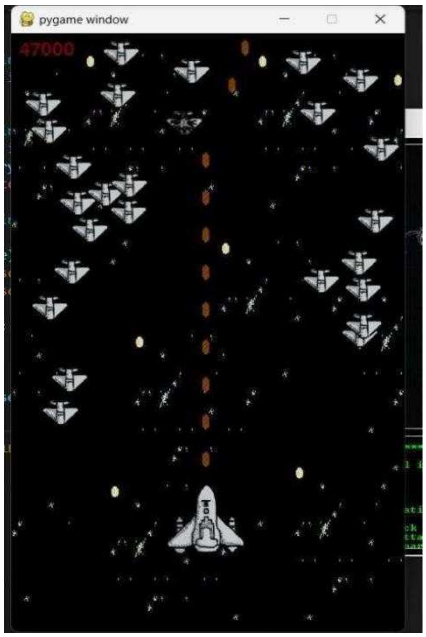

Fig. 10. playing Space Shooting game using our VPA

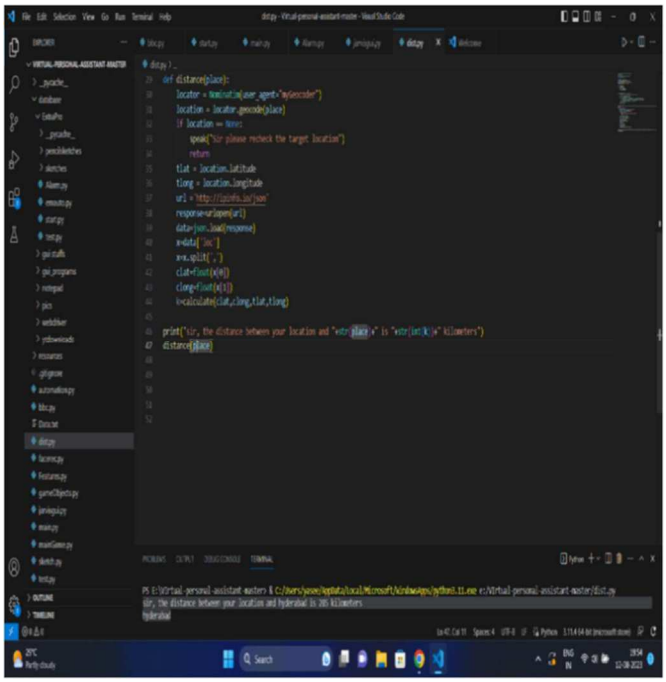

Fig. 11. calculating distance between user's location given location

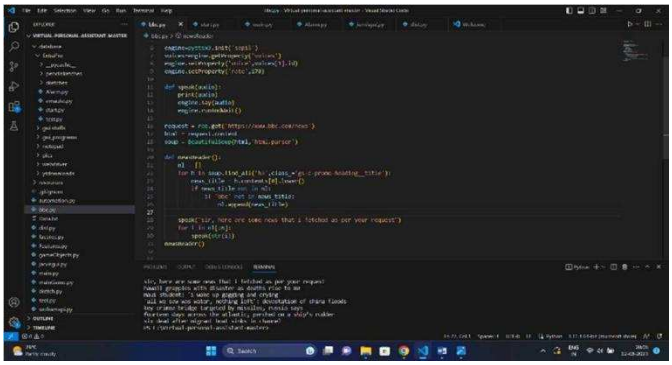

Fig. 12. News Reader feature using IntelliNeo

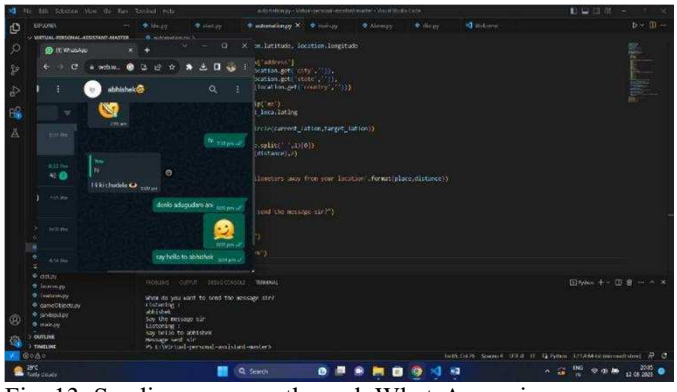

Fig. 13. Sending message through WhatsApp using IntelliNeo

V. CONCLUSION

Voice assistants can only perform certain tasks up to a certain level, but when it comes to higher level of automation tasks, they are unable to perform well. Therefore, we have developed our application in a way

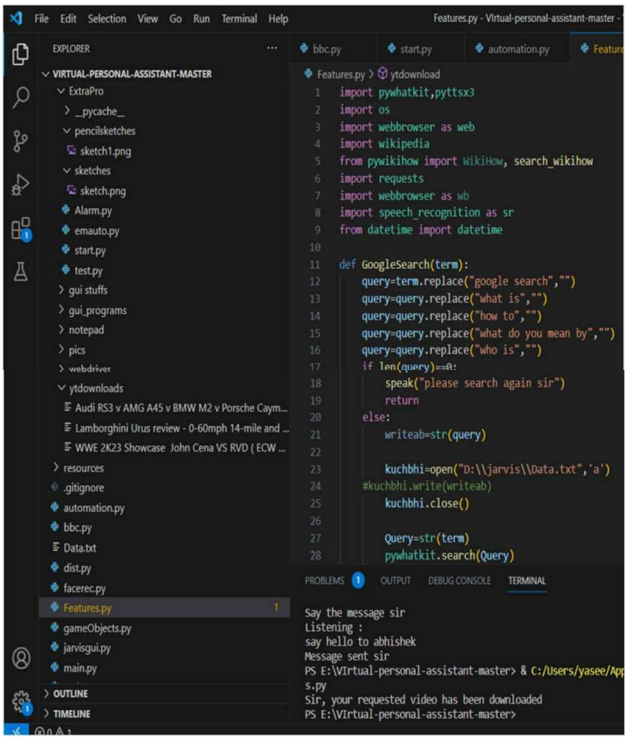

Fig. 14. download videos from YouTube

that would enable them to perform tasks at a level where those in need could effectively use them, such as YouTube automation, News Scrapper, checking some information which is wanted by the user and performing all the tasks. Hence, we have been successfully created part of the infrastructure needed for voice assistants, and in future we can expect our virtual assistants to be in the form of holographic image and behave more similar to a human brain and perform powerful task far beyond human intelligence.

REFERENCES

[1] Aabhas Kumar, Damandep Kaur, Abhishek Kumar Pathak, "Voice assistant using python", 2022 International Conference on Cyber Resilience (ICCR) — 978-1-6654-6122-1/22/ ©2022 IEEE —DOI: 10.1109/ICCR56254.2022.9995997

[2] Shailaja Uke and et al, "Virtual voice assistant using python (friday)", 2022 IEEE 4th International Conference on Cybernetics, Cognition and Machine Learning Applications (978-1-6654-6246-4/22/©2022 IEEE—DOI: 10.1109/ICCCMLA56841.2022.9989099

[3] Abhishek Joshi, Shaik Vaseem Akram, "Personalized desktop app based interactive means of zira voice assistant" 2022 International Interdisciplinary Humanitarian Conference for Sustainability (IIHC) 978-1-6654-5687-6/22/©2022IEEE—DOI: 10.1109/IIHC55949.2022.10060610

[4] Bhawana Sati and et al, "An Intelligent Virtual System using Machine Learning", 2022 IEEE IAS Global Conference on Emerging Technologies (GlobConET)— 978-1-6654-4357-9/22/ ©2022 IEEE — DOI: 10.1109/GlobConET53749.2022.9872396

[5] Ravivanshikumar Sangpal, Tanvee Gawand, Sahil Vaykal, Neha Madhavi, "Jarvis: An interpretation of AIML with integration of gTTS and Python", 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)

[6] Vadaboyina Appalaraju, V Rajesh, K SaiKumar, P. Sabitha, K Ravi Kiran, "Design and development of Intelligent voice personal assistant using python", 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)

[7] Satyendra Praneel Reddy Karri, B. Santhosh Kumar, "Deep learning techniques for implementation of chatbots", 2020 International Conference on Computer Communication and Informatics (ICCCI - 2020), Jan 22 – 24, 2020, Coimbatore, INDIA

[8] Regina Gubareva, Rui Pedro Lopes, “Virtual assistants for learning: a systematic literature review”

[9] D RajkumarPillay and et al, “Implementing an artificial intelligence based ideal form of virtual personal assistant dsign for various communication medium”, Proceedings of the Third International Conference on Electronics and Sustainable Communication Systems (ICESC 2022) IEEE Xplore Part Number: CFP22V66-ART; ISBN: 978-1-6654-7971-4

[10] Prajyot Mane, Shubham Sonone, Nachiket Gaikwad, “Smart personal assistant using machine learning, International Confer- ence on Energy, Communication, Data Analytics and Soft Computing” (ICECDS-2017)

[11] Rahul Kumar, Garima Sarupria, Varshil Panwala, Smit Shah, Nehal Shah, “Power efficient smart home with voice assistant”, 2020 11th In- ternational Conference on Computing, Communication and Net- working Technologies (ICCCNT), 2020, pp. 1-5, doi: 10.1109/ICC-CNT49239.2020.9225612.

[12] S Akash, Neeraj Jayaram, “Creating Desktop Speech Recognization Using Python Programming”. IJARCCE, Vol. 10, Issue 3, March 2021, ISSN (Online), pp.129-134.

[13] Nil Goksel, Mehmet Emin Mutlu, “On the track of artificial intelligence: Learning with intelligent personal assistants”. Journal of Human Sciences, 13(1):592–601, 2016

[14] Muneeb Ali Hameed, Yaser Hafeez, Bushra Hameed, Mamoona Humayun, Noor Zaman Jhanjhi, “Towards an effective approach for architectural knowledge management considering global software development”. International Journal of Grid and Utility Computing, 11(6):780–791, 2020

[15] P. Krishnaraj, F.Mohamed Faris, D. Rajesh, “Portable voice recognition with gui automation” 2021, IJIRT, Volume 9 Issue 6 ISSN (Print): 2320-9356 PP. 20-23.