Developing Python Based Voice Assistant And Implementing Prebuild Packages In Application

P.A. Harley

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY (HONS.)

(MECHATRONICS ENGINEERING)

SUBMITTED BY :

ROHAN DEB MONDAL 1801306008
NAVEEN CHAUDHARY 1801306006

QUANTUM UNIVERSITY, ROORKEE

May 2022

Developing Python Based Voice Assistant And Implementing Prebuild Packages In Application

P.A.- Harley

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY (HONS.)

(MECHATRONICS ENGINEERING)

SUBMITTED BY :

ROHAN DEB MONDAL 1801306008

NAVEEN CHAUDHARY 1801306006

QUANTUM SCHOOL OF TECHNOLOGY

ROORKEE

QUANTUM UNIVERSITY, ROORKEE

May 2022

DECLARATION

We hereby certify that the work which is being presented in the project entitled "**P.A-HARLEY**" by "**ROHAN DEB MONDAL and NAVEEN CHAUDHARY**" in partial fulfilment of requirements for the award of the degree of B.Tech(Hons.) (Mechatronics Engineering) submitted in the Department of Mechanical and Mechatronics Engineering at **QUANTUM SCHOOL OF TECHNOLOGY, ROORKEE** affiliated to **QUANTUM UNIVERSITY, ROORKEE** is carried out during a period from JULY 2021 to MAY 2022 under the supervision of "**NAVEEN RANA, ASS. PROF.**". The matter presented in this project has not been submitted by us to any other University / Institute for the award of B. Tech. degree of :

1. **ROHAN DEB MONDAL**
2. **NAVEEN CHAUDHARY**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

NAVEEN RANA
ASS. PROFESSOR
GUIDE

ABSTRACT

In today's world, "ALEXA" is very popular. People want a better life and as much automation as possible than a virtual assistant that meets the needs of their daily lives and responds well to loneliness. Personalization is a concern and has a profound impact, whether at a self-satisfaction level or a business level. At our level, we are very passionate about technology and automation. Moreover, automation has been a blessing to lazy engineers like us, and if it happens on an individual level, it enhances the experience even more. Even in the business sector, 76% of consumers choose brands that they personalize. Of those, 78% are more likely to recommend and repurchase to friends and family.

In that spike of personalisation, another popular Hollywood movie Iron Man sparked the thought of 'Jarvis'- a personalised voice-assisted AI virtual assistant. And in the toughest of times and future world of technology and automation an AI virtual assistant looks like a family member to every household in near future.

We both were sitting together ideal and listening to other mates using Alexa and we had no money in hand then. So worked over repositories and all possible open libraries and jotted down our own version of virtual assistant inspired by the DC character Harley Quinn.

Our project basically dives into simple but robust use of python functions and simpler package and functionality inputs and tried to be simple thus everyone can implement and accordingly modify the codes and build their own version of a personal assistant.

We have tried to implement both online and offline desktop implications and also integrated simple integrations to implement personalisation.

The main purpose of the assistant is to be responsive and smart toward the speech by using the natural language processing package 'speechrecognition' used and also responding in voice and displaying text. The assistant also in connection to internet services can deliver www services. It converts the voice into a computer-readable language to give the required solution to the answer or deny if unknown.

The assistant is work ready and can be implemented on ground level.

ACKNOWLEDGEMENT

This letter of thanks is just a note or documented format of our acknowledgement towards all the persons involved in this project be it physically, mentally or emotionally. But in reality, no words are enough to express our sincere gratitude to them. With respect, we acknowledge our project to our project guide and beloved teacher all through our college, **Naveen Rana, Ass. Professor, Quantum School of Technology**, without whose guidance and support we may not have been able to successfully bring up the results we wanted. He has always been supportive and even guided us at the document stage of the project.

Our heartfelt thanks to **Dr Manish Sharma, Director, Quantum School of Technology, Roorkee**, for providing this opportunity to carry out the project. Our constant guide and encourager **Er. M. Kannan, Sr.Ass. Professor & Head, Department of Mechatronics Engineering,** has been of great help in carrying out the project and we acknowledge him reverential thanks.

We thank and respect all our faculty members of the ME/MTE department and the CSE department for their regular intellectual support on this project.

Our love and regards for our family for their trust and support as always been throughout this project and life.

Above all the great almighty GOD who made this possible to happen and may his blessing be upon all of the worlds.

Rohan Deb Mondal

Naveen Chaudhary

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Introduction

In this age of technology, all that humans can do is be replaced by machines. One of the main causes is its performance changes. In today's world, we train machines to think like humans, and the machine/robot does the job. Thus, the concept of a virtual assistant was born.

Virtual assistants generally use a speech recognition function and a language processing algorithm to recognize the user's given voice commands and perform related actions according to the user's request (online or offline). It is a digital assistant that performs based on the specific commands provided by the user, the virtual assistant can filter out the ambient noise and return relevant information. Although the Virtual Assistant is completely software-based, it is now integrated into a variety of devices, and there are also some assistants designed specifically for individual brand ecosystems, such as Alexa and Siri. With upcoming advancements in AI(Artificial Intelligence), deep learning, machine learning and advanced NLP(Natural Language Processing) in today's world it's practical that we train machines with these technologies. Even it has given a big boost to the voice assistant industry. And voice assistant makes us talk with our machines and that is a far better communication parameter than being able to just chat and visualise with machines.

1.2 Importance of Virtual Assistants

These assistants are pretty useful for aged, blind & physically challenged people, children, etc. by making the interaction with the machine not a challenge anymore for people. Even blind people who could never see the machine can interact and operate it using their voice only. Daily based tasks such as reading headlines, getting a weather update, setting an alarm, playing music, notifying, etc seem like a token of words now with these assistants. We have developed a voice assistant that works on both Linux and Windows platforms. It is based on python modules and libraries. Our assistant can do all the basic functions but with Machine Learning, IoT and AI implementation this can be taken to do wonders. But due to budget limitations and to keep it simple and user friendly so that anyone can implement it along for personal modifications we have kept it simple. Python modules and libraries are used in this project implementing machine learning to train the model, and some

os commands are also added to test on os capabilities. By implementing machine learning and deep learning we can easily implement upon the usage to which the assistant is assigned. With the help of Voice Assistant, there will be no need to write the commands again and again for performing particular tasks. Once the model is created it can be used any number of times by any number of users in the easiest ways. So, with the help of the virtual assistant, we will be able to control many things around us single-handedly on one platform without the struggle of even moving from a place or type but by the comfort of our own mouths.

1.3 Working Principle

Voice assistant applications work based on the Automatic Speech Recognition (ASR) system. ASR systems record the speech and make small tokens of phonemes, which are later get processed into text and then implemented. A phoneme (not words of syllables) is a basic unit of measurement for human speech recognition.

Being the fact that speech is a primary form of communication, the growth of speech technology is an important step toward controlling unstructured voice data. To limit and control the power of speech, and bettering lives with speech technology, Machine Learning (ML) plays a vital role in developing the many speech processing algorithms. Over the past year, Alexa has learned how to take over references from one query to the next and to deliver follow-up questions without having to repeat the wake word. Further essentially, with deep learning, we're able to model a large number of domains and transfer that learning to a new domain or skill.

Artificial intelligence gives modern voice assistant apps the freedom from limited vocabulary but uses cloud storage with millions of words and phrases instead. Speech processing and Natural Language Processing (NLP) using ML allow intelligent devices, such as smartphones, to interact with users via verbal language. Thus opening a huge scope for this project in future.

Personal use voice assistants such as Siri, Google Home, and Amazon Alexa are devices that offer individualized speech technology experiences. These Intelligent Personal Assistants (IPAs) made our day-to-day lives easy and more comfortable. Smart businesses never stop the search for a better user experience for their end consumers. Voice recognition technology is all about getting more out of the users of the device they already own. Big companies like Apple, Google, Amazon, and Microsoft are doing a remarkable job to improve their voice assistants.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

Virtual Personal Assistant, or VPA, is software that acts as a virtual choreographer that manages other services, bringing them together in the most effective manner as per our use. Many of us already work with primary versions of Virtual Personal Assistants today, when we use Google Now, Apple's Siri, Amazon's Alexa, Microsoft's Cortana, etc. The upcoming advancements are to bring about major improvements and companies are extensively working and developing the AIML and VPA industry.

In a few years, VPA services taking over unwanted sections in industries and homes won't be a surprise. The moment we step out of the house with family for a tour VPA will inform the local grocery vendor to stop delivery of groceries, reschedule existing appointments, etc will be common in upcoming days. Virtual Personal Assistants have two basic functions: they understand us and our context and use that understanding to develop other software to do things on our behalf. VPAs act as mediators, or agents, and in order to be effective in that eventually we will need to know them closely. The major six building blocks make up a Virtual Personal Assistant that can be categorised as user understanding, contextual input and service choreography.

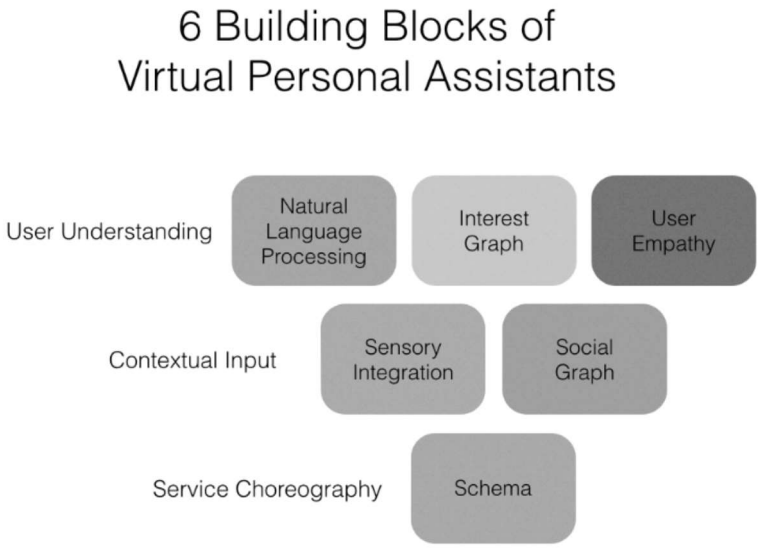

Fig. 1. Six building blocks of VPA

2.2 Natural Language Processing

Natural Language Processing (NLP) refers to the computer science department — and in particular, the branch of artificial intelligence or AI — which deals with giving computers the ability to understand the text and spoken words in the same way that human beings can. NLP combines computer languages — the official modelling of human languages — with mathematics, machine learning, and deep learning models. Combined, these technologies allow computers to process human language in the form of text or voice data and 'understand' their complete, complete meaning for the speaker or author's purpose.

NLP runs computer programs that translate text from one language to another, respond to spoken commands, and summarize large documents quickly — even in real-time. There are plenty of opportunities for you to connect with NLP through GPS voice systems, digital assistants, speech and text call software, customer service chatbots, and other consumer useful features. But NLP also plays a growing role in business solutions that help simplify business operations, increase employee productivity, and simplify the most important business processes.

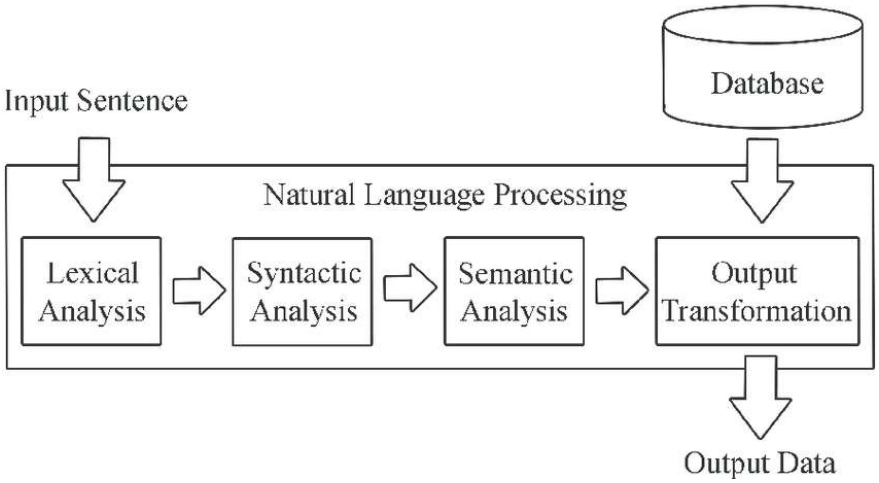

Fig. 2. NLP steps

Several NLP activities break human text and voice data in ways that help a computer make sense of what it is importing. Some of these activities include the following:

- Speech recognition, also called speech-to-text, it is a function of faithfully converting voice data into text data. Speech recognition is required for any app that follows voice commands or answers spoken questions. What makes speech recognition

particularly challenging is the way people speak — quickly, combining words, various accents and pronunciation, with different pronunciations, and often using the wrong grammar.

- Part of speech tagging, also called grammatical tagging, is a process of determining the part of speech of a particular word or piece of text based on its use and condition. Part of the speech identifies 'make' as a verb in 'We can make a VPA in python.' and as a noun in 'What make of bike do you ride?'
- Word sense disambiguation is the selection of the correct meaning of any word with several meanings through a process of semantic analysis that determine the word that makes the most sensible meaning in the given situation.
- Named entity recognition, or NEM identifies words or phrases as useful tokens. NEM identifies 'Mumbai' as a location or 'Naveen' as a man's name.
- Co-reference resolution, is a function of pointing out that two words refer to the same thing or person. For ex- Rohan is fat like a bear. So bear contexts to a man, not an animal here.
- Sentiment analysis attempts to extract possible subjective qualities—attitudes, emotions, sarcasm, confusion, etc from the text.
- Natural language generation is the opposite of speech recognition or speech-to-text. It's like putting structured text to human language form.

The Python programing language provides a wide range of tools and libraries for attacking specific NLP tasks. Many of these are found in the Natural Language Toolkit, or NLTK, an open-source collection of libraries, programs, and education resources for building NLP programs.

2.3 Interest Graph

One way of thinking about a 'graph' is like a map of the relationship between objects. In the case of a 'Interest graph', it is drawn up on our relationships and interests. The interest graph puts a map of our interests.

Virtual Personal Assistants will be the main way we handle interest graphs. Today, the way we do that is by interacting with hundreds of websites and services, each of which maintains a unique, personalized profile of the interests we express through that particular service. Internet leaves our fingerprints on everything we touch, especially as application designers are seen giving us quick and easy ways to express our interests seamlessly as we move forward with resources.

Our distributed online profiles take many forms, including patterns we display on our likes on Facebook, our search on Google, and our purchases on Amazon.
As the Visual Assistant handles our additional interactions with various websites and services, our interest graph is compiled within VPA.

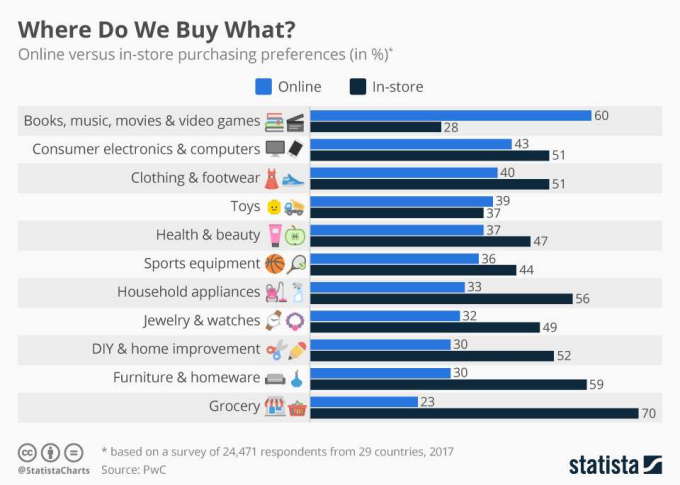

Fig. 3. Interest Graph Online vs. In-store

2.4 User Empathy

We deceive ourselves into believing that we are rational people, but the truth is that our emotions strongly influence our knowledge of the world. An emotionally dumb VPA will never work for us again as an emotionally intelligent person.
The challenge here is to read accurately the feelings of the other person, and there are two ways to do so today. The first is to analyze the text with tools like Linguistic Inquiry and Word Count (LIWC), which Facebook has instead misused to identify and suppress negative sentiment in a survey of about 700,000 of its users. LIWC only works with large pieces of text (over 400 words), however, which may limit its effectiveness in many of our interactions with VPA.
The most possible way of VPA emotional intelligence is finding one's emotions through speech analysis and video-based facial analysis. Companies working in this space, such as Affectiva, Emotient and NICE, are small firms, focused on call center applications and solutions to test the emotional responses of advertising. In this case, Affectiva is particularly interesting. It has grown out of MIT Media Lab and is developing mobile solutions to provide in-device emotional processing. I would be surprised if they were soon taken over by one of VPA's biggest players.

2.5 Sensory Integration

Sensors now enable us to measure location, ambient light, air pressure, humidity, temperature and a range of other sensory data right on our phones. Thus they provide vital physical environments, and that aids understanding the state of the situation i.e, our sensory perceptions being technically upgraded.

Visual Assistants will not necessarily need to understand our speech, our interests and our feelings; by providing them with direct feeds from advanced sensor broadcasts, they will help us better understand what is happening around us. Are we within walking distance of a delightful KFC outlet at the time of our breakfast? What are the current movies trending. Is there a movie I was awaiting for?

Thus external sensor inputs improves understanding of the situation and integration with IoT can do wonders with sensor implementation. Like when the temperature rises too high turn on the A.C. automatically, or derive cost expenses from electricity usage.

Home appliances	Power (W)	Hours of use per day	Energy consumed per day
Fridge	400	24	9.60
Light room 1	80	8	0.64
Light room 2	80	5	0.40
Light room 3	80	8	0.64
Light room 4	80	8	0.64
Light room 5	80	6	0.48
TV 1	70	6	0.42
TV 2	70	6	0.42
Total (Kwh)			13.24
Cost Kwh ($)			0.09
Total to pay Daily ($)			1.19

Tab. 1. Data generated by virtual assistant integrated with IoT on electrical expense per day over the appliances used

And with camera integration and facial recognition even moods of the user can be tracked and interpreted in advance terms.

"Contextual computing" is a term used to describe how the upcoming expansion of nerves will help us to better navigate our world. In essence, through its integration of sensors, the VPA will play a key role in encapsulating our physical and virtual realities.

2.6 Social Graph

Much has been done in the last decade of Facebook's growth and growing importance of the "social graph" - a map of our interactions with other people. Nowadays most of the world has a working internet connection and a smartphone. This outbreak is even visible in the rising numbers of users getting added in social media. Instagram, whatsapp, twitter, tiktok, etc are in high spikes in growth rates. In recent days business platforms are shifting business to social platforms to generate more leads.

Within the context of the Virtual Person Assistant, the social graph is similar to the sum of the senses; just that instead of mapping our physical condition, the map is our 'social context' - and that is the most important thing in human society nowadays. Much of what we do with our VPA, we will do together with other people, whether it be sharing a ride, having a coffee, or going to a conference.

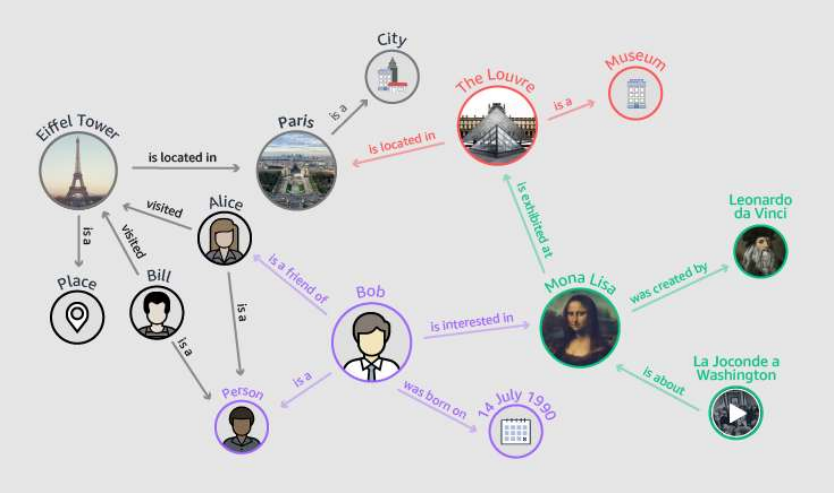

Fig. 4. Social Mapping AWS Neptune

The social graph tells the VPA who matters to us – and by extension guides its coordination with other people's VPAs.

2.7 Schema Integration

Once the VPA understands what we need, the next step is to connect those needs to third-party web services. The VPA's ability to understand human communication will be extremely complex, but it cannot be imagined that the same level of professionalism exists with its service partners. To ensure clear communication, both parties need to agree on

some form of standard language. In information science, that common language is called an "ontology.". In software development, it is called a schema.

The schemes will play a key role in allowing third-party service developers to send and receive VPA to and fro communications. Schema.org, a collaboration between Google, Microsoft, Yahoo and Yandex, is a noteworthy schema. Facebook with OpenGraph schema.

Facebook focuses on sports, media, restaurants and fitness, while Schema.org has a wide range of in-depth interesting information on events, creative activities, healthcare, organization types, locations, bookings and product descriptions.

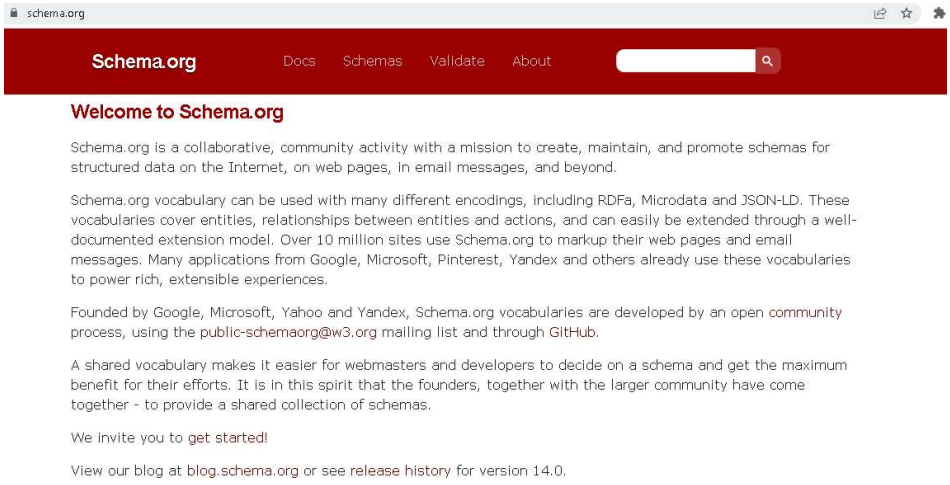

Fig. 5. Schema.org website

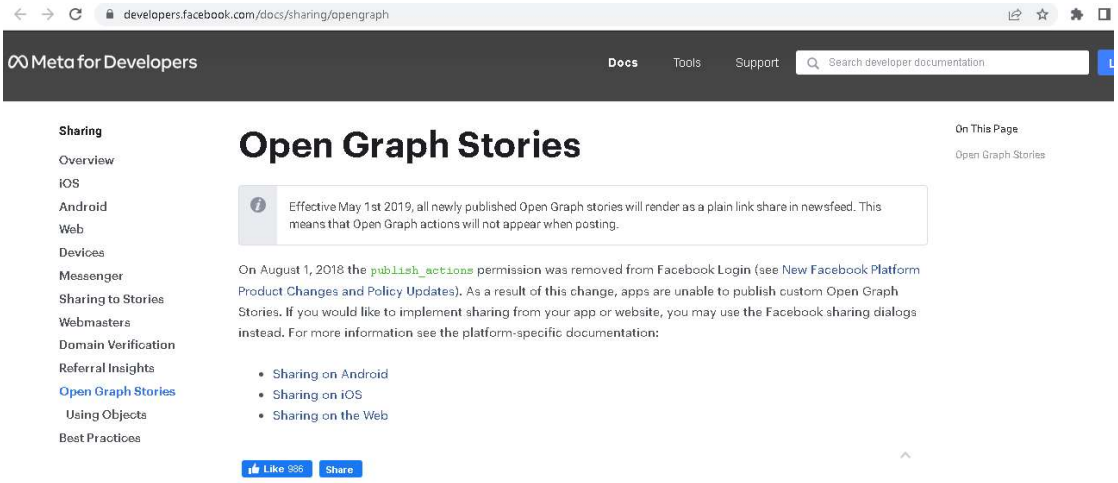

Fig. 6. Open Graph website

Chapter 3

PRESENT WORK- P.A. HARLEY

3.1 Introduction

Our project began in our hostel room itself. One day we both sitting in our room heard our friend using Alexa. As we didn't have enough money and we were just in wfh environment that time. Salary was to arrive after training. We were finding ways to make ourselves a way to enjoy our virtual assistant. So we deep dived the internet and started revising our beloved python and its libraries that we studied in out second year. Started implementing smaller functions and going through intensive trials until we ended up completing our final year project- Harley, our own personal assistant.

It's simple, robust, can understand and respond to basic tasks, web api also included so weather information, etc are also under it's capabilities. Anyone can implement our code and improve itds limitations and implications.

3.2 Python

"Python is powerful... and fast; plays well with others; runs everywhere; is friendly & easy to learn; is Open. - Python.org

For more than 20 years, Python has been used successfully throughout the world as a programming language in industry, service, and research and science to meet a wide range of needs. During this time, the language has changed many things. Python is easy to learn language. It's removing the boundaries between users and developers. Growing numbers of scientists, engineers, financiers, professionals and others with little planning experience uses Python to solve some complex technical problems.

"Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library."- Wikipedia

Guido van Rossum began working on Python in the late 1980s as a replacement for ABC programming language and first releasing it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list recognition, cycle-based garbage collection, reference calculations, and Unicode support. Python 3.0, released in 2008, was

a major overhaul of the latest version. Python 2 was discontinued after version 2.7.18 by 2020.

Python is intended to be an easy-to-read language. Its formatting looks plain and often uses English keywords when other languages use tough terms and also punctuation as end of statement. Unlike many other languages, it does not use curved brackets to separate blocks, and semicolons after statements are allowed but are rarely used. It uses indentation instead. It has fewer syntactic variants and special cases than C or Pascal.

3.3 Getting Started with Python

Python source code and installers are available for download for all versions.

One can easily download platform specific download files to install python from:

https://www.python.org/downloads/

There are several python integrated development environments(IDEs) available across the internet. For example Jupiter Notebook, Visual Studio Code, etc.

Fig. 7. IDLE- python shell editor

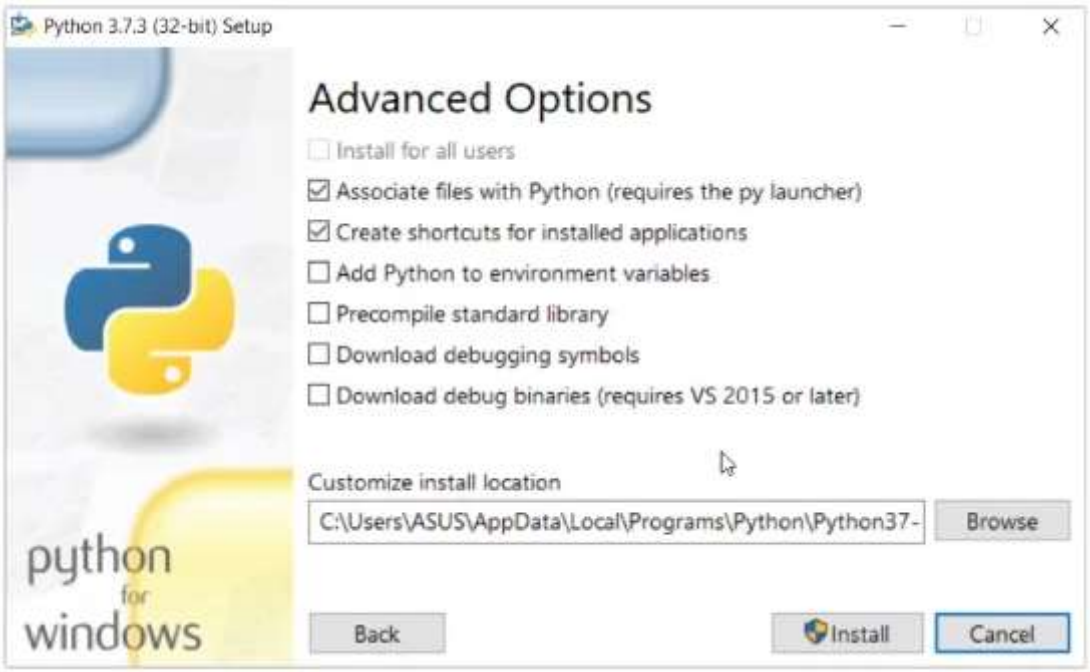

Fig. 8. Python Installation Window

We have used Microsoft Visual Studio Code for our project. It's simple working with visual code. For basic introduction the below link can be referred to:

https://code.visualstudio.com/docs/python/python-tutorial

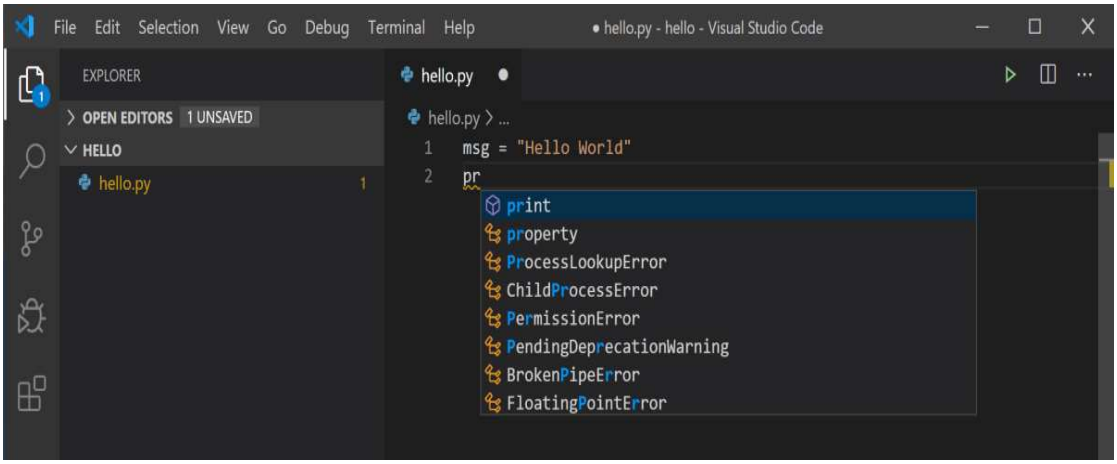

Fig. 9. Vs Code- Python

- First Program: Hello, World program syntax in python,

  //prints Hello, World in console

  print("Hello, World")

3.4 Virtual Environment and Packages

Python applications will often use packages and modules that do not come as part of a standard library. Apps will sometimes require a special version of the library, as the application may require some error correction or the application may be written using an outdated version of the library interface.

This means it may not be possible for one Python installation to meet the requirements of every application. If application A needs version 1.0 of a particular module but application B needs version 2.0, then the requirements are in conflict and installing either version 1.0 or 2.0 will leave one application unable to run.

The solution to this problem is to create a virtual environment, a standalone directory tree that contains Python installation for a specific version of Python, and a number of additional packages.

❖ Creating virtual environment:

The module used to create and manage virtual environments is called **venv**. venv will usually install the most recent version of Python that user have available.

To create a virtual environment, decide upon a directory where you want to place it, and run the venv module as a script with the directory path:

```
$ python3 -m venv tutorial-env
```

This will create a tutorial-env directory if not available, and also create directories within it that contains a copy of the Python translator and various supporting files.

The default directory for the visible directory is .venv. This name keeps the list hidden in your shell and thus goes off track while providing a name that explains why the directory exists. It also prevents conflicts with .env environment variable definition files that some tooling supports.

Once you've created a virtual environment, you may activate it.

On Windows, run:

```
$ tutorial-env\Scripts\activate.bat
```

On Unix or MacOS, run:

```
$ source tutorial-env/bin/activate
```

To verify if the environment has been activated or not, you can see (tutorial-env) in your terminal.

❖ Managing packages with pip:

We can install, upgrade, and remove packages using a program called pip. By default pip will install packages from the Python Package Index, <https://pypi.org>. We can browse the Python Package Index by going to it in your web browser.

We can install the latest version of a package by specifying a package's name-

```
(tutorial-env) $ python -m pip install novas
```

We can also install a specific version of a package by giving the package name followed by == and the version number-

```
(tutorial-env) $ python -m pip install requests==2.6.0
```

We can run pip install --upgrade to upgrade the package to the latest version:

```
$ python -m pip install --upgrade requests
```

3.5 Libraries used

In this project we have used simple yet powerful libraries that help us implement this project. Libraries used in this project are listed below:

- pyttsx3: pyttsx is a cross-platform text to speech library which is platform-independent. The major advantage of this library in text-to-speech conversion is that it works offline. To install this module, type the command in the terminal-

  ```
  $ pip install pyttsx3
  ```

- SpeechRecognition: Library for performing speech recognition-functionalities to convert audio into text for further processing or operations, with support for several engines and APIs, online and offline. Type the below command in the terminal-

  ```
  $ pip install SpeechRecognition
  ```

- pywhatkit: PyWhatKit is a Python library with various helpful features that will help us interact with the browser very easily. Being easy-to-use and no hidden setup. Use the below command:

  ```
  $ pip install pywhatkit
  ```

- wikipedia: We have used this to fetch a variety of information from the Wikipedia website. To install this module, type the below command in the terminal:

  ```
  $ pip install wikipedia
  ```

- requests: Requests is a simple, yet elegant, HTTP library. It allows us to send HTTP/1.1 requests extremely easily.

To download requests module enter the code in terminal:

$ pip install requests

- Python-decouple: Decouple helps you to organize your settings so that you can change parameters without having to redeploy your app. It also makes it easy for you to:
    - store parameters in ini or .env files;
    - define comprehensive default values;
    - properly convert values to the correct data type;
    - have only one configuration module to rule all your instances.
    - For installation type in the command below in console:

$ pip install python-decouple

We also create a .env file to store some private data such as API Keys, Passwords, and so on that are related to the project.

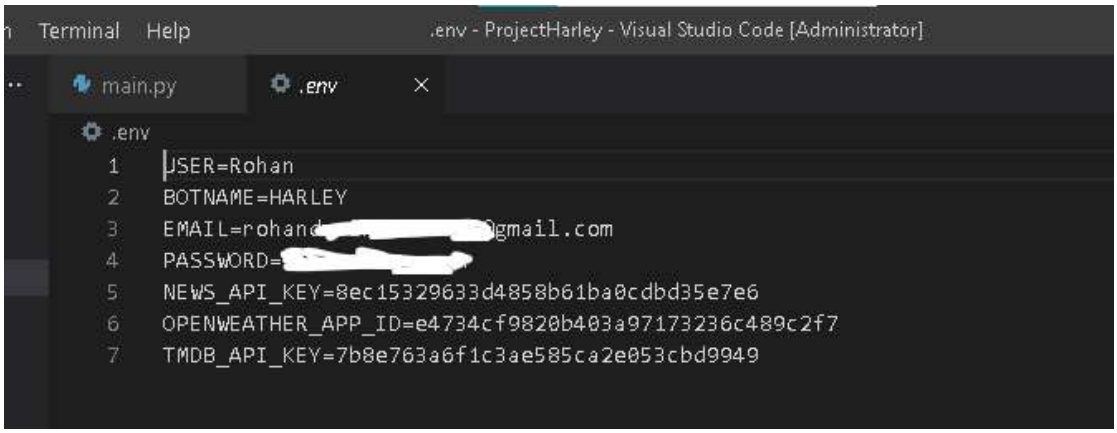

Fig. 10. .env file contents

3.6 Problem Statement and Final Project Folder

The initial spark of 'Alexa' gave rise to a problem statement of making a Voice Assistant that is personalised, but problem statement with few boxes to check:

- ☑ ~~Simple, can be modified according to personal choice~~
- ☑ ~~Should work in multiple platforms~~
- ☑ ~~Should be robust and must be voice actioned~~
- ☑ ~~Should be able to perform basic greeting to simple on OS operations~~
- ☑ ~~Should atleast be able to perform basic web tasks as well~~

And after checking all the boxes, we were ready with our working code. We had initially faced a lot of issues with the formatting and interfacing. Issues with voice capture and pyaudio. But fixing all the bugs and issues we completed our project folder.

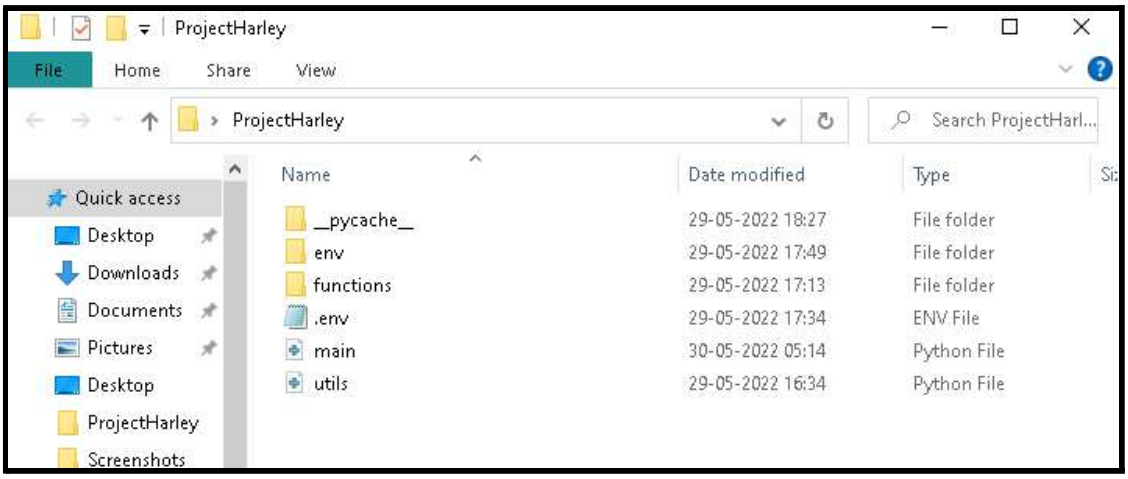

Fig. 11. Final Project Folder

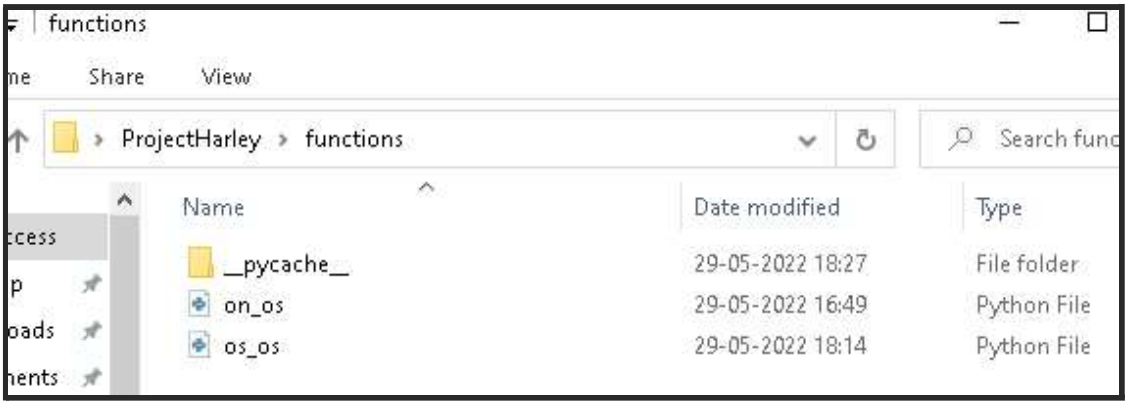

Fig. 12. functions Folder

GitHub Repository Link to the project:

https://github.com/rdm-99/P.A.Harley.git

For any help related to the repository or files contact: qwertyromo@gmail.com

Chapter 4

RESULT AND DISCUSSION

We successfully debugged all the errors and we come to the final code. We have a main driver code that deals with all the package calls and function calls. We have a utils file that contains the opening text. We have a functions folder that contain the online functionalities and in os functionalities.

Final Python Files:

main.py

```
1.  import requests
2.  import pyttsx3
3.  import speech_recognition as sr
4.  from decouple import config
5.  from datetime import datetime
6.  from functions.on_os import get_random_joke, find_my_ip, get_random_advice,
    get_latest_news, get_trending_movies, get_weather_report, play_on_youtube,
    search_on_google, search_on_wikipedia, send_email, send_whatsapp_message
7.  from functions.os_os import open_cmd, open_calculator, open_camera,
    open_notepad, open_discord
8.  from random import choice
9.  from utils import opening_text
10. from pprint import pprint
11.
12.
13. USERNAME = config('USER')
14. BOTNAME = config('BOTNAME')
15.
16.
17. engine = pyttsx3.init('sapi5')
18.
19. # Setting speech Rate
20. engine.setProperty('rate', 170)
```

```
21.
22.  # Setting the Volume
23.  engine.setProperty('volume', 100.0)
24.
25.  # Setting Voice Tone(Female)
26.  voices = engine.getProperty('voices')
27.  engine.setProperty('voice', voices[1].id)
28.
29.
30.  # Text to Speech Conversion
31.  def speak(text):
32.      """speaks whatever text is passed to it"""
33.
34.      engine.say(text)
35.      engine.runAndWait()
36.
37.
38.  # Greet the user
39.  def greet_user():
40.      """Greets the user according to the time currently"""
41.
42.      hour = datetime.now().hour
43.      if (hour >= 6) and (hour < 12):
44.          speak(f"Good Morning joke, {USERNAME}")
45.      elif (hour >= 12) and (hour < 16):
46.          speak(f"Good afternoon {USERNAME}")
47.      elif (hour >= 16) and (hour < 19):
48.          speak(f"Evening time, cookies time {USERNAME}")
49.      else:
50.          speak(f"Hello,{USERNAME}, Is it dark outside? My time, right?")
51.      speak(f"I am {BOTNAME} from Suicide Squad. What do you want from
     Harley?")
52.
53.
```

```
# Takes Input from User
def take_user_input():
    """user input, recognizes it using Speech Recognition module and converts it into text"""

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print('Listening....')
        r.pause_threshold = 1
        audio = r.listen(source)

    try:
        print('Recognizing...')
        query = r.recognize_google(audio, language='en-in')
        if not 'exit' in query or 'stop' in query:
            speak(choice(opening_text))
        else:
            hour = datetime.now().hour
            if hour >= 21 and hour < 6:
                speak("Good night joke, take care!")
            else:
                speak('Have a good day! keep hunting bats')
            exit()
    except Exception:
        speak('Sorry, could you speak what I understand. Could you repeat like that batsy?')
        query = 'None'
    return query


if __name__ == '__main__':
    greet_user()
    while True:
        query = take_user_input().lower()
```

```
86.
87.         if 'open notepad' in query:
88.             open_notepad()
89.
90.         elif 'open discord' in query:
91.             open_discord()
92.
93.         elif 'open command prompt' in query or 'open cmd' in query:
94.             open_cmd()
95.
96.         elif 'open camera' in query:
97.             open_camera()
98.
99.         elif 'open calculator' in query:
100.            open_calculator()
101.
102.        elif 'ip address' in query:
103.            ip_address = find_my_ip()
104.            speak(f'Joke your IP is {ip_address}.\n For your sins, I am printing it on the
    screen as well.')
105.            print(f'Your IP Address is {ip_address}')
106.
107.        elif 'wikipedia' in query:
108.            speak('What is there so intersting on Wikipedia, joke?')
109.            search_query = take_user_input().lower()
110.            results = search_on_wikipedia(search_query)
111.            speak(f"According to Wikipedia, {results}")
112.            speak("For your fun, I am printing it on the screen joke.")
113.            print(results)
114.
115.        elif 'youtube' in query:
116.            speak('What do you want to play on Youtube, is it on harley sir?')
117.            video = take_user_input().lower()
118.            play_on_youtube(video)
```

```
119.
120.         elif 'search on google' in query:
121.             speak('What do I say Google to search for, sir?')
122.             query = take_user_input().lower()
123.             search_on_google(query)
124.
125.         elif "send whatsapp message" in query:
126.             speak(
127.                 'Whom should I send the message sir? Please enter in the console: ')
128.             number = input("Enter the number: ")
129.             speak("What is the message you wish to convey?")
130.             message = take_user_input().lower()
131.             send_whatsapp_message(number, message)
132.             speak("I've sent the message joke.")
133.
134.         elif "send an email" in query:
135.             speak("Tell me the email? will you? Please enter in the console: ")
136.             receiver_address = input("Enter email address: ")
137.             speak("What should be the subject joke?")
138.             subject = take_user_input().capitalize()
139.             speak("And, your message?")
140.             message = take_user_input().capitalize()
141.             if send_email(receiver_address, subject, message):
142.                 speak("I've sent the mail.")
143.             else:
144.                 speak("Something went wrong while I was sending the mail. Please check
    the error logs joke")
145.
146.         elif 'joke' in query:
147.             speak(f"Why so serious? joker wants a joke...")
148.             joke = get_random_joke()
149.             speak(joke)
150.             speak("I am printing it on the screen. laugh on it")
151.             pprint(joke)
```

```
152.
153.         elif "advice" in query:
154.             speak(f'Joke wants an advice from harley. Ok tell you some")
155.             advice = get_random_advice()
156.             speak(advice)
157.             speak("I am printing it on the screen.")
158.             pprint(advice)
159.
160.         elif "trending movies" in query:
161.             speak(f'Suicide Squad please do watch and other trending movies are:
    {get_trending_movies()}")
162.             speak("I am also printing it on the screen joke.")
163.             print(*get_trending_movies(), sep='\n')
164.
165.         elif 'news' in query:
166.             speak(f'Reading funny and laughable, oops! sorry latest news headlines")
167.             speak(get_latest_news())
168.             speak("You can also find it on screen")
169.             print(*get_latest_news(), sep='\n')
170.
171.         elif 'weather' in query:
172.             ip_address = find_my_ip()
173.             city = requests.get(f"https://ipapi.co/{ip_address}/city/").text
174.             speak(f'Getting weather over your city {city}")
175.             weather, temperature, feels_like = get_weather_report(city)
176.             speak(f'The current temperature is {temperature}, but it feels like
    {feels_like}")
177.             speak(f'Also, as it says {weather}")
178.             speak("See the screen")
179.             print(f'Description: {weather}\nTemperature: {temperature}\nFeels like:
    {feels_like}")
180. //end of main.py
```

utils.py

```
opening_text = [
    "Funny, I'm on it joke.",
    "Okay Harley is working on it.",
    "Just a moment joke.",
]
//end of utils.py
```

Inside the functions folder:

on_os.py

```
import requests
import wikipedia
import pywhatkit as kit
from email.message import EmailMessage
import smtplib
from decouple import config

NEWS_API_KEY = config("NEWS_API_KEY")
OPENWEATHER_APP_ID = config("OPENWEATHER_APP_ID")
TMDB_API_KEY = config("TMDB_API_KEY")
EMAIL = config("EMAIL")
PASSWORD = config("PASSWORD")


def find_my_ip():
    ip_address = requests.get('https://api64.ipify.org?format=json').json()
    return ip_address["ip"]


def search_on_wikipedia(query):
    results = wikipedia.summary(query, sentences=2)
    return results

```

```
def play_on_youtube(video):
    kit.playonyt(video)


def search_on_google(query):
    kit.search(query)


def send_whatsapp_message(number, message):
    kit.sendwhatmsg_instantly(f"+91{number}", message)


def send_email(receiver_address, subject, message):
    try:
        email = EmailMessage()
        email['To'] = receiver_address
        email["Subject"] = subject
        email['From'] = EMAIL
        email.set_content(message)
        s = smtplib.SMTP("smtp.gmail.com", 587)
        s.starttls()
        s.login(EMAIL, PASSWORD)
        s.send_message(email)
        s.close()
        return True
    except Exception as e:
        print(e)
        return False


def get_latest_news():
    news_headlines = []
    res = requests.get(
```

```
58.
    f"https://newsapi.org/v2/top-headlines?country=in&apiKey={NEWS_API_KEY}&cat
    egory=general").json()
59.     articles = res["articles"]
60.     for article in articles:
61.         news_headlines.append(article["title"])
62.     return news_headlines[:5]
63.
64.
65. def get_weather_report(city):
66.     res = requests.get(
67.
    f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={OPENWEATHE
    R_APP_ID}&units=metric").json()
68.     weather = res["weather"][0]["main"]
69.     temperature = res["main"]["temp"]
70.     feels_like = res["main"]["feels_like"]
71.     return weather, f"{temperature}°C", f"{feels_like}°C"
72.
73.
74. def get_trending_movies():
75.     trending_movies = []
76.     res = requests.get(
77.
    f"https://api.themoviedb.org/3/trending/movie/day?api_key={TMDB_API_KEY}").jso
    n()
78.     results = res["results"]
79.     for r in results:
80.         trending_movies.append(r["original_title"])
81.     return trending_movies[:5]
82.
83.
84. def get_random_joke():
85.     headers = {
```

```
86.         'Accept': 'application/json'
87.     }
88.     res = requests.get("https://icanhazdadjoke.com/", headers=headers).json()
89.     return res["joke"]
90.
91.
92. def get_random_advice():
93.     res = requests.get("https://api.adviceslip.com/advice").json()
94.     return res['slip']['advice']
95. //end of on_os.py
```

os_os.py

```
1.  import os
2.  import subprocess as sp
3.
4.  paths = {
5.      'notepad': "C:\\Windows\\system32\\notepad.exe",
6.      'discord': "C:\\Users\\qwert\\AppData\\Local\\Discord\\app-1.0.9004\\Discord.exe",
7.      'calculator': "C:\\Windows\\System32\\calc.exe"
8.  }
9.
10.
11. def open_notepad():
12.     os.startfile(paths['notepad'])
13.
14.
15. def open_discord():
16.     os.startfile(paths['discord'])
17.
18.
19. def open_cmd():
20.     os.system('start cmd')
21.
22.
```

```
23. def open_camera():
24.     sp.run('start microsoft.windows.camera:', shell=True)
25.
26.
27. def open_calculator():
28.     sp.Popen(paths['calculator'])
29. //end of os_os.py
```

No Errors Generated and Working Output:

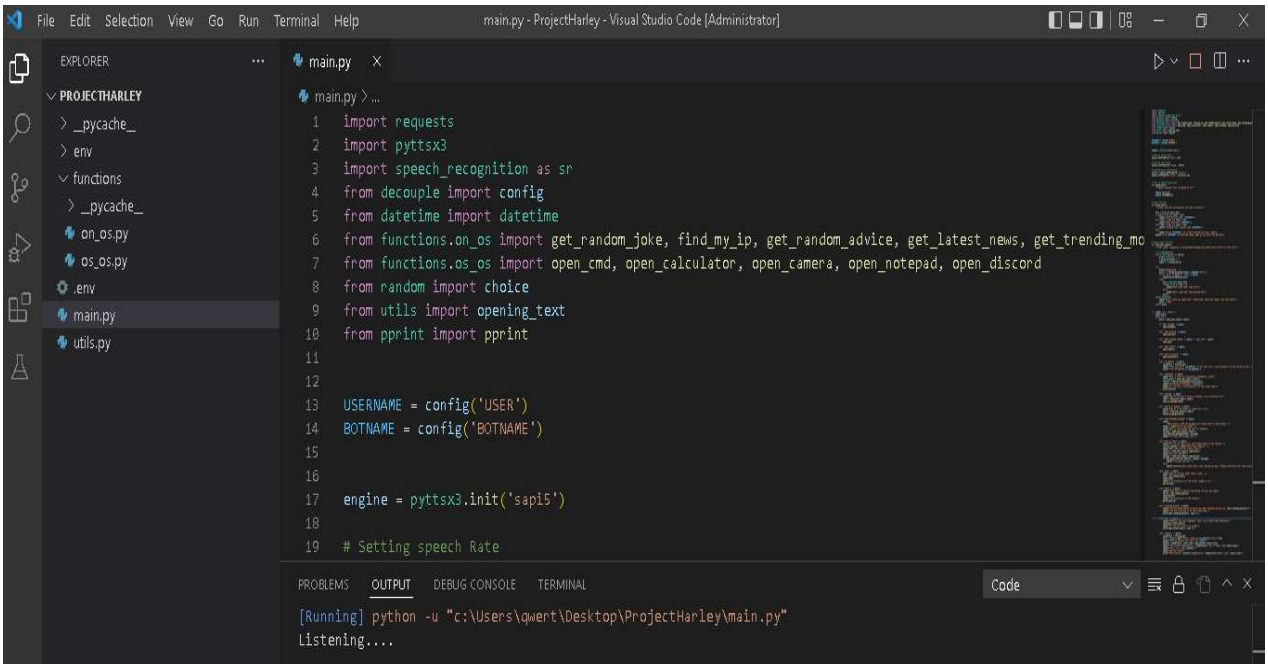

Fig. 13. Output

Thus, the project is successfully implemented and is debugged at all stages and is ready to implement.

Errors Guide:

For environmental variables or virtual environment related issues refer:

https://python.plainenglish.io/do-you-really-need-environment-variables-in-python-201f4abd46b8

Issues related to Microsoft Speech API(SAPI) refer:

https://docs.microsoft.com/en-us/previous-versions/windows/desktop/ee125663(v=vs.85)

Issue related to pyaudio refer:

https://github.com/benfred/implicit/issues/371

https://stevemats.medium.com/solved-fix-pyaudio-pip-installation-errors-on-a-win-32-64-bit-operating-system-1efe6cd90c8d

IP address fetching error API refer:

https://www.ipify.org/

For News API key:

https://newsapi.org/

For Weather API key:

https://openweathermap.org/

For Trending movies API key:

https://www.themoviedb.org/

For Python 3.10 and above Python Match Case can be used instead of if-elif ladder. For tutorial refer:

https://peps.python.org/pep-0636/

Any Python related query or tutorial refer:

python.org

To run the program, you can use the following command:

$ python main.py

CONCLUSION AND FUTURE SCOPE

Hence, we successfully made our own personalised voice-enabled Virtual Assistant- Harley. We utilised open-source python packages and learnt about the packages as well. How they work, though many implements were out of our scope of study but yet having those insights and their use in practical use. Python is so quick to learn but yet so deep and vast, and with regular updates if you are ready to grasp in the new and explore out of your bounds, it serves as one of the most powerful tools.

Python has machine learning, AI, deep learning implications, document implications, web based implications, app-platform level development, developing games,etc. With proper knowledge and knowing where to apply. Knowing the basics of coding and algorithms build the base to any program. Same for python, more we dive in it gives us endless opportunities and parameters. We learned several new packages and also came across the various functions enabled with it.

As per current industry parameters, implementing this VPA with sensors and IoT and do some modifications in its AI applications, we can actually have a Smart Home, Smart Office implementation. Where voice and AI be the power can serve a great economy boost as well as faster and better interaction.

For example in a smart phone, the vpa deals with all the regular work like switching lights off and on as per light intensity and state of people in room. Turn on A.C. or fan according to ambient temperature and presence of bill. Auto open and lock door based on facial recognition by vpa. Calculating and updating the expenses and average monthly electricity bills and intelligently save electricity by turning off lights and unnecessary appliances not in use. Fill pet food plate automatically as per pet schedule when user out of home. When out of city, auto inform grocery to not deliver the products as per regular schedule, etc.

Thus future implications of this project is feasible and is in high demand and soon to be a reality with upcoming improvements in AI, virtual reality and other sensor and processing techniques. “The JARVIS magic is reality….

As we come to the end of this project, our heartfelt thanks to all the well wishers and all the support throughout the project.

REFERENCES

- Python.org- referred for several tutorials and issues
  1. https://docs.python.org/3/tutorial/index.html
  2. https://docs.python.org/3/tutorial/controlflow.html#function-annotations
  3. https://docs.python.org/3/tutorial/controlflow.html#unpacking-argument-lists
  4. https://docs.python.org/3/tutorial/modules.html#importing-from-a-package

- For Creation of virtual environments: https://docs.python.org/3/library/venv.html
- For speech recognition package: https://pypi.org/project/SpeechRecognition/
- For WhatsApp automation package: https://pypi.org/project/pywhatkit/
- Wikipedia:
  - package: https://pypi.org/project/wikipedia/
  - As ref: https://en.wikipedia.org/wiki/Python_(programming_language); https://en.wikipedia.org/wiki/Python_(programming_language)#API_documentation_generators; https://en.wikipedia.org/wiki/Python_(programming_language)#Libraries
- PyPI requests package: https://pypi.org/project/requests/
- https://pypi.org/help/- For pip issues took reference
- NLP: https://www.ibm.com/cloud/learn/natural-language-processing
- VS code: https://code.visualstudio.com/docs/?dv=win
- API Links:
  - https://newsapi.org/docs
  - https://openweathermap.org/
  - https://www.themoviedb.org/documentation/api
  - https://api.adviceslip.com/
- Flask Error: https://exerror.com/importerror-no-module-named-flask/
- py audio wheel: https://www.lfd.uci.edu/~gohlke/pythonlibs/#pyaudio
- Article on AI and Voice Assistant: https://medium.com/swlh/ai-revolution-voice-assistants-their-smartness-c7afe2580e74
- Research papers referred:
  - https://www.researchgate.net/publication/351723449_A_Voice_Based_Assistant_Using_Google_Dialogflow_and_Machine_Learning

    - https://www.researchgate.net/publication/264001644_Virtual_Personal_Assistant
- Earlier project referred: https://extrudesign.com/virtual-assistant-using-python/
- Personalisation importance: https://www.mckinsey.com/business-functions/growth-marketing-and-sales/our-insights/the-value-of-getting-personalization-right-or-wrong-is-multiplying
- CSAT based on AI based VPA results: https://www.researchgate.net/publication/337256309_Siri_Alexa_and_other_digital_assistants_a_study_of_customer_satisfaction_with_artificial_intelligence_applications
- IIT past research paper cum project journal: https://ijirt.org/Article?manuscript=152099
- 3 categories of machine learning: https://www.guavus.com/ai-vs-machine-learing-vs-data-mining-whats-big-difference-part-2/
- Article on VPA smartness: https://medium.com/swlh/ai-revolution-voice-assistants-their-smartness-c7afe2580e74
- https://www.defined.ai/case-studies/building-a-voice-assistant-model/

Above all references are true to my knowledge and have been used as reference for this project idea, design and implementation. Any reference missed here is an unfortunate outcome and not intentionally done.

Thank You…