Table of Contents

[**INTRODUCTION:** 5](#_Toc120874344)

[**MOTIVATION ABOUT THIS PROJECT:** 6](#_Toc120874345)

[**PROBLEM DEFINITION:** 7](#_Toc120874346)

[**OBJECTIVE OF THE PROJECT:** 8](#_Toc120874347)

[**EXISTING SYSTEM AND PROPOSED SYSTEM:** 9](#_Toc120874348)

[**PROPOSED SYSTEM** 10](#_Toc120874349)

[**FUNCTIONAL REQUIREMENTS:** 10](#_Toc120874350)

[**SOFTWARE REQUIREMENTS:** 11](#_Toc120874351)

[**HARDWARE REQUIREMENTS:** 12](#_Toc120874352)

[**SOFTWARE:** 13](#_Toc120874353)

[**MODULES USED IN THIS PROJECT:** 14](#_Toc120874354)

[**SPEECH RECOGNITION** 14](#_Toc120874355)

[**Pyttsx3** 14](#_Toc120874356)

[**PyAutoGUI** 15](#_Toc120874357)

[**WIKIPEDIA** 15](#_Toc120874358)

[**WHERE THE AI USED NOW-A-DAYS AND IN FUTURES** 16](#_Toc120874359)

[ **AI for Marketing** 16](#_Toc120874360)

[ **AI for Improved Operations in Businesses** 16](#_Toc120874361)

[ **AI for Recruitment** 16](#_Toc120874362)

[ **AI for Smart Homes** 16](#_Toc120874363)

[ **Artificial Intelligence for E-commerce** 16](#_Toc120874364)

[ **Artificial Intelligence for Education** 16](#_Toc120874365)

[**FIGMA** 16](#_Toc120874366)

[**MACHINE LEARNING WITH PYTHON:** 17](#_Toc120874367)

[**SENTIMENT ANALYSIS:** 17](#_Toc120874368)

[**ARCHITECTURE DESIGN:** 19](#_Toc120874369)

[**UML DIAGRAMS:** 20](#_Toc120874370)

[**CLASS DIAGRAM:** 20](#_Toc120874371)

[**USE-CASE DIAGRAM:** 21](#_Toc120874372)

[**ACTIVITY DIAGRAM:** 21](#_Toc120874373)

[**DATAFLOW DIAGRAM:** 22](#_Toc120874374)

[**SEQUENCE DIAGRAM:** 23](#_Toc120874375)

[**COMPONENT DIAGRAM:** 23](#_Toc120874376)

[**DEPLOYMENT DIAGRAM:** 24](#_Toc120874377)

[**What is SDLC?** 26](#_Toc120874378)

[**Stage 1: Planning and Requirement Analysis** 27](#_Toc120874379)

[**Stage 2: Defining Requirements** 27](#_Toc120874380)

[**Stage 3: Designing the Product Architecture** 27](#_Toc120874381)

[**Stage 4: Building or Developing the Product** 28](#_Toc120874382)

[**Stage 5: Testing the Product** 28](#_Toc120874383)

[**Stage 6: Deployment in the Market and Maintenance** 28](#_Toc120874384)

[**SDLC Models:** 28](#_Toc120874385)

[**PHASES OF SDLC:** 29](#_Toc120874386)

[**SAMPLE CODE:** 31](#_Toc120874387)

[**(Alpha.py)** 31](#_Toc120874388)

[**INTRODUCTION:** 40](#_Toc120874389)

[**Types of Manual Testing** 41](#_Toc120874390)

[**1. White Box Testing** 41](#_Toc120874391)

[**2. Black Box Testing** 41](#_Toc120874392)

[**3. Grey Box Testing** 41](#_Toc120874393)

[**How to perform Manual Testing** 42](#_Toc120874394)

[**SCREENSHOTS:** 44](#_Toc120874395)

[**CONCLUSION:** 46](#_Toc120874396)

[**FUTURE SCOPE:** 47](#_Toc120874397)

[**Websites Referred:** 48](#_Toc120874398)

[**Websites Referred:** 48](#_Toc120874399)

[**YouTube Referred:** 48](#_Toc120874400)

[**Blog or Documentation Referred:** 48](#_Toc120874401)

|  |  |  |
| --- | --- | --- |
| **S.NO** | **FIGURE TITLE** | **Page No** |
| 1 | Microsoft Cortana | 9 |
| 2 | Speech Recognition | 14 |
| 3 | Figma | 16 |
| 4 | Neural Network | 17 |
| 5 | Architecture design | 19 |
| 6 | Class diagram | 20 |
| 7 | Use case diagram | 21 |
| 8 | Activity diagram | 21 |
| 9 | Data flow diagram | 22 |
| 10 | Sequence diagram | 23 |
| 11 | Component diagram | 23 |
| 12 | Deployment diagram | 24 |
| 13 | Waterfall model | 29 |
| 14 | Output screen | 43 |
| 15 | Output screen | 44 |
| 16 | Output screen | 45 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TABLE TITLE** | **Page No** |
| 1 | Test case of this AI Personal Assistant | 42 |

**LIST OF TABLE**

**AI PERSONAL ASSISTANT**

# **INTRODUCTION:**

The Virtual Assistant started in 1910s, and the growth of technology has supported virtual Assistant improvement. The application of Artificial Intelligence (AI) also was a turning point in virtual Assistant journey. Using AI to develop the Virtual Assistant was a great jump to increase the Virtual Assistant capabilities. Currently Virtual Assistant use narrow AI with limited option. However, using general AI in the near future can be revolution to improve the quality of Virtual Assistant services. A Virtual Assistant, also called AI assistant or digital assistant, is an important program that understand Natural language voice commands and completes tasks for the user.

The Artificial Intelligence-Powered Virtual Assistant uses advanced Intelligence. It is a piece of software that understand verbal and written command and complete the task assigned by the client.

Some of the famous assistant may we use in our daily life for daily purpose like: Alexa, Siri, Google Assistant, Bixby and Microsoft Cortana. All this AI Assistant make the human life year how the technology is increasing now-a-days all the human work are doing robot or AI and automating the robot to do the work.

## **MOTIVATION ABOUT THIS PROJECT:**

The complexity of the emotional presentation of users to Artificial Intelligence (AI) virtual assistants is mainly manifested in user motivation and social emotion, but the current research lacks an effective conversion path from emotion to acceptance. This paper innovatively cuts from the perspective of trust, establishes an AI virtual assistant acceptance model, conducts an empirical study based on the survey data from 240 questionnaires, and uses multilevel regression analysis and the bootstrap method to analyse the data.

The results showed that functionality and social emotions had a significant effect on trust, where perceived humanity showed an inverted U relationship on trust, and trust mediated the relationship between both functionality and social emotions and acceptance. The findings explain the emotional complexity of users toward AI virtual assistants and extend the transformation path of technology acceptance from the trust perspective, which has implications for the development and design of AI applications.

## **PROBLEM DEFINITION:**

Virtual Assistant can give answers to the question and perform the specific task that command by the user. It can’t control the system using the user voice command and can’t automate the user work and etc. This AI Personal Assistant can listen your word and it will respond and perform the task and automate the user work and user can control the system using voice command and open the application and perform the different types of tasks etc. This AI Assistant use the productivity purpose also. This application is built for desktop/laptop.

We used several case studies for each student about distraction, procrastination, privacy, security, usability, and more in the learning process. Student can use this assistant to ask the question and get their answers without touching the system. We can control the whole system using our voices.

## **OBJECTIVE OF THE PROJECT:**

This application is designed for desktop and laptop. We are trying to build the AI assistant using some advance using neural network and other more features and functions.

The application is written In a Visual Studio and this project uses Python- programming language. The main purpose of an intelligent virtual assistant is to answer questions that users may have. This may be done in a business environment, for example, on the business website, with a chat interface. On the mobile platform, as in the case of Apple's Siri the intelligent virtual assistant is available as a call-button operated service where a voice asks the user “What can I do for you?” and then responds to verbal input.

Beyond the core utilities of an intelligent virtual assistant, companies are now exploring how to enhance it further. One main element of this involves adding personality; for example, some suggest that, by "consolidating" various efforts, the IT community could build more highly advanced intelligent virtual assistants with more developed personalities and capabilities.

**SYSTEM ANALYSIS**

## **EXISTING SYSTEM AND PROPOSED SYSTEM:**

**EXISTING SYSTEM**

Currently many popular AI assistants are present include Amazon Alexa, Apple's Siri, Google Assistant and Microsoft's Cortana.

Figure 1:- Microsoft cortana...

Another commonly understood AI assistant is Microsoft’s Cortana. To give customized suggestions, Cortana uses the Bing search engine, NLP, and data from smartphones. It has APIs that interact with different apps from Windows and 3rd parties. The functions Cortana performs include:

* Hands-free assistance
* Questions for answers
* Provides reminders and retains notes
* Takes charge of assignments
* Helps in calendar management

## **PROPOSED SYSTEM**

* This application is building for desktop.
* Control the system application using voice command module.
* Automating the desktop to control using user command.
* you can modify and change the function as per the user needs.
* Every AI have their important role but this AI role is make human life easier and studies.
* Some features of the AI Assistant:

1. Greeting us
2. Search Wikipedia
3. Search anything on Google or YouTube
4. Operating system related operation
5. Make Notes
6. Telling latest new
7. Control System

## **FUNCTIONAL REQUIREMENTS:**

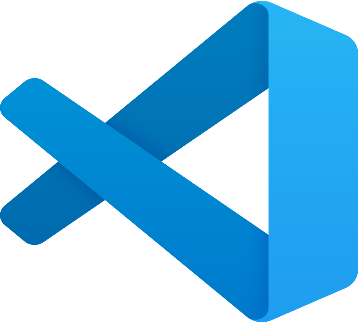
Requirements analysis in systems engineering and software engineering encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, such as beneficiaries or users. A software requirements specification (SRS) is a document that is created when a detailed description of all aspects of the software to be built must be specified before the project is to commence. It is important to note that a formal SRS is not always written.

In fact, there are many instances in which effort expended on an SRS might be better spent in other software engineering activities. Requirements analysis is critical to the success of a development project. Requirements must be actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. By analysing different hardware components and familiar software following are hardware & software used in our project.

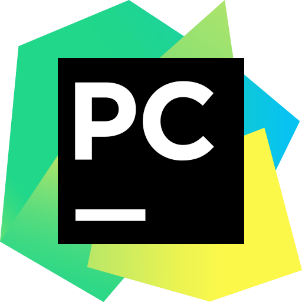
### **SOFTWARE REQUIREMENTS:**

1. Operating System: Windows 11 (64-bit)
2. Programming Language: Python (3.11.0)
3. Any Integrated Development Environment (IDE) or Code Editor.

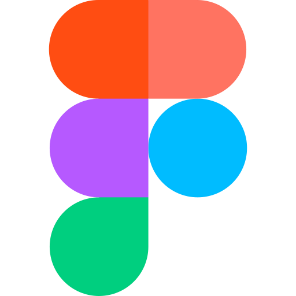
Example: Microsoft Visual Studio.



**Microsoft Visual Studio:** Visual Studio Code is a code editor in layman’s terms. Visual Studio Code is “a free-editor that helps the programmer write code, helps in debugging and corrects the code using the intelligence-sense method”. In normal terms, it facilitates users to write the code in an easy manner. Many people say that it is half of an IDE and an editor, but the decision is up to the coders. Any program/software that we see or use works on the code that runs in the background. Traditionally coding was used to do in the traditional editors or even in the basic editors like notepad! These editors used to provide basic support to the coders.



**PyCharm:** PyCharm is compatible with Linux, macOS, and Windows architectures, usable as a multi-platform program. PyCharm is one of the better IDEs in Python and supports the two versions of Python 2 (2.7) and Python 3 (3.5 and higher).  PyCharm provides a host of plugins, kits, and tools to accelerate [Python development](https://www.educba.com/what-is-python/) and simultaneously minimize the effort taken to achieve the same. In addition, PyCharm can be tailored according to production needs and personal preferences.

**Figma:** Figma is a collaborative web application for interface design, with additional offline features enabled by desktop applications for macOS and Windows. The feature set of Figma focuses on user interface and user experience design, with an emphasis on real-time collaboration, utilising a variety of vector graphics editor and prototyping tools. The Figma mobile app for Android and iOS allows viewing and interacting with Figma prototypes in real-time on mobile and tablet devices.

### **HARDWARE REQUIREMENTS:**

1. RAM : 8GB or More
2. PROCESSOR : i3 8th gen or higher
3. ROM : 100GB or more
4. PROGRAMMING LANGUAGE : python 3.11

**SOFTWARE ENVIRONMENT**

# **SOFTWARE:**

**PYTHON:**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast.

# **MODULES USED IN THIS PROJECT:**

# **SPEECH RECOGNITION**

Speech recognition is a machine's ability to listen to spoken words and identify them. You can then use speech recognition in [Python](https://www.simplilearn.com/learn-the-basics-of-python-article) to convert the spoken words into text, make a query or give a reply. You can even program some devices to respond to these spoken words. You can do speech recognition in python with the help of computer programs that take in input from the microphone, process it, and convert it into a suitable form.

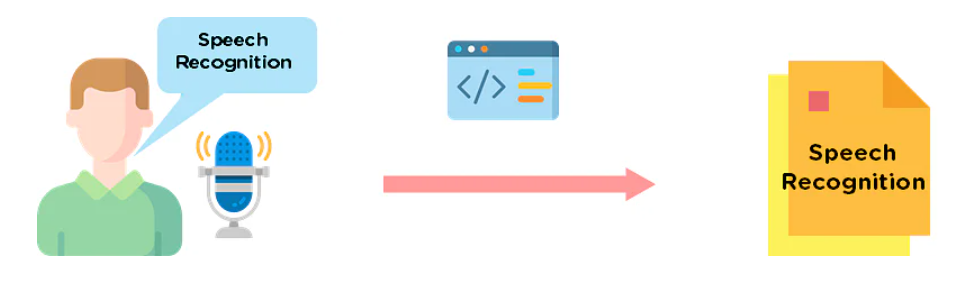
Speech recognition seems highly futuristic, but it is present all around you. Automated phone calls allow you to speak out your query or the query you wish to be assisted on; your virtual assistants like Siri or Alexa also use speech recognition to talk to you seamlessly.

Figure 2:- Speech recognition

# **Pyttsx3**

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech. The pyttsx3 module supports two voices first is female and the second is male which is provided for windows.

# **PyAutoGUI**

PyAutoGUI lets your Python scripts control the mouse and keyboard to automate interactions with other applications. The API is designed to be simple. PyAutoGUI works on Windows, macOS, and Linux, and runs on Python 2 and 3.

# **WIKIPEDIA**

The Internet is the single largest source of information, and therefore it is important to know how to fetch data from various sources. And with Wikipedia being one of the largest and most popular sources for information on the Internet.

Wikipedia is a multilingual online encyclopedia created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system.

In this article, we will see how to use Python’s Wikipedia module to fetch a variety of information from the Wikipedia website**.**

**Is AI in demand?**

Artificial intelligence is one of the most in-demand skills in the world. But what is it about AI that employers are so eager to hire? There are a few key reasons: first, AI is versatile. It can be used in a variety of industries, from finance to healthcare. Second, AI is constantly evolving. There are new applications and opportunities for AI being discovered all the time. Third, AI is becoming more affordable. With the rise of cloud computing and open-source AI software, businesses of all types are embracing the technology of Artificial Intelligence.

Artificial intelligence is in demand across different industries because it is widely seen as an enhancer of human productivity. For instance, more and more companies are starting to look at AI to replace human labour in certain aspects of their business. AI is also very lucrative in the marketing world, where it is used to gain insight into user behaviour and help in targeted promotions. But the biggest attribute of AI that makes it so popular is that it is highly scalable, and the possibilities are endless. You can only do so much with human talent at any given time, but you can make AI learn whatever you want it to learn, theoretically speaking at least.

# **WHERE THE AI USED NOW-A-DAYS AND IN FUTURES**

## **AI for Marketing**

## **AI for Improved Operations in Businesses**

## **AI for Recruitment**

## **AI for Smart Homes**

## **Artificial Intelligence for E-commerce**

## **Artificial Intelligence for Education**

# **FIGMA**

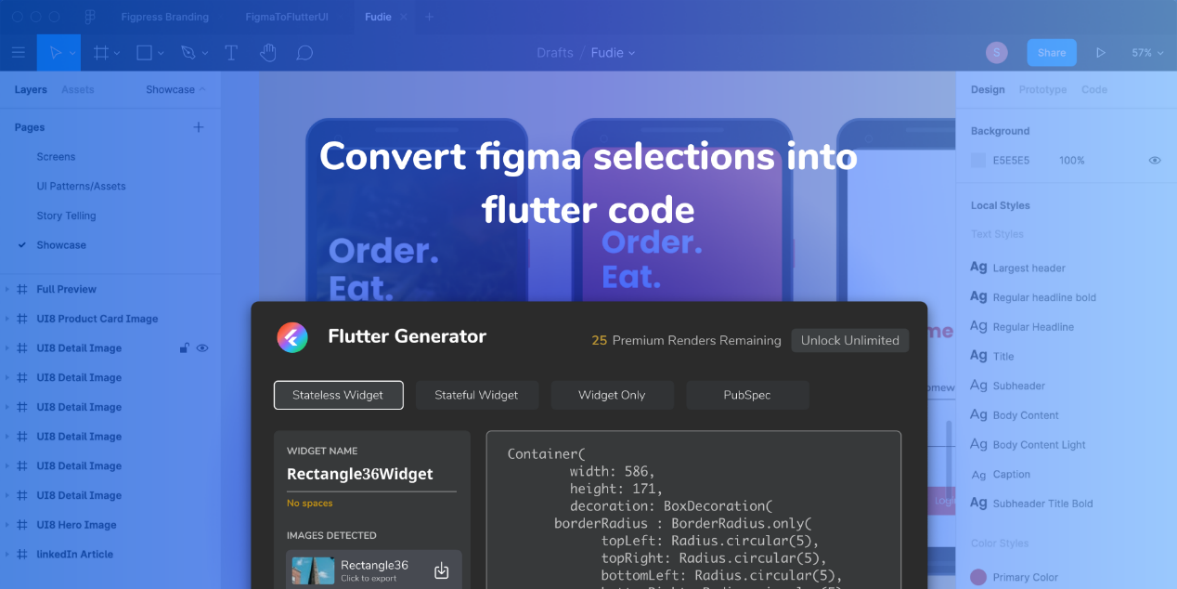
Figma is a collaborative interface design tool that’s taking the design world by storm. Unlike Sketch, which runs as a standalone MacOS app, Figma is entirely browser-based, and therefore works not only on Macs, but also on PCs running Windows or Linux, and even on Chromebooks. It also offers a web API, and it’s free!

Figure 3:- Figma

Another big advantage of Figma is that it allows real-time collaboration on the same file. When using conventional “offline” apps like Sketch and Photoshop, if designers want to share their work, they typically have to export it to an image file, then send it via email or instant message.

In Figma, instead of exporting static images, we can simply share a link to the Figma file for clients and colleagues to open in their browser. This in itself saves significant time and inconvenience in a designer’s workflow. But more importantly, it means that clients and colleagues can interact more richly with the work, and review the latest version of the file.

## **MACHINE LEARNING WITH PYTHON:**

### **SENTIMENT ANALYSIS:**

Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.



Figure 4:- Neural Network

Sentiment analysis is a machine learning tool that analyses texts for polarity, from positive to negative. By training machine learning tools with examples of emotions in text, machines automatically learn how to detect sentiment without human input.

To put it simply, machine learning allows computers to learn new tasks without being expressly programmed to perform them. Sentiment analysis models can be trained to read beyond mere definitions, to understand things like, context, sarcasm, and misapplied words. For example:

***“Super user-friendly interface. Yeah right. An engineering degree would be helpful.”***

Out of context, the words ‘super user-friendly’ and ‘helpful’ could be read as positive, but this is clearly a negative comment. Using sentiment analysis, computers can automatically process text data and understand it just as a human would, saving hundreds of employee hours.

Imagine using machine learning to process customer service tickets, categorize them in order of urgency, and automatically route them to the correct department or employee. Or, to analyse thousands of product reviews and social media posts to gauge brand sentiment.

**SYSTEM DESIGN & UML DIAGRAM**

# **ARCHITECTURE DESIGN:**

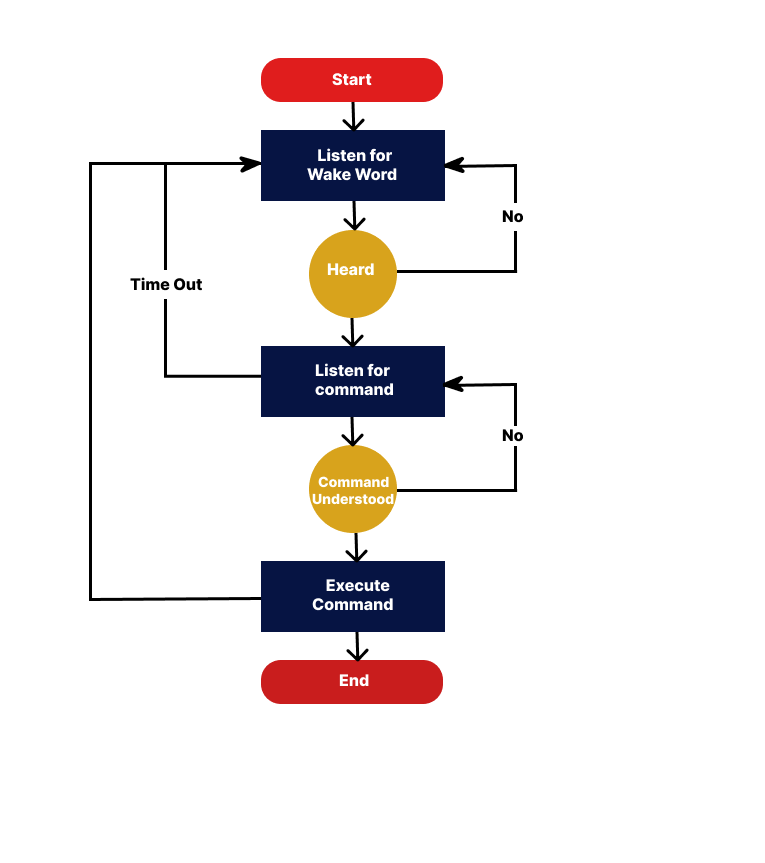
****

Figure 5:- Architecture Diagram of this AI Personal Assistant

It will start Listening and give the voice command to the AI personal assistant. If the AI assistant heard the voice command it go the next step if not will go back to the start position and we have to say that again to the assistant. After listening the voice command it will recognize it in certain period of time if not. It will be time out and we have to repeat again. After Understanding the voice command go to next step. Use voice command will be executed successfully. Over, Repeat the step for another command.

# **UML DIAGRAMS:**

The model consists of user input through microphone to accept commands from the user. These commands are then go through Speech Recognition, it is the ability of a machine or program to identify words and phrases in spoken languages and convert them to a machine-readable format. On these input Natural Language Processing is applied, it is a field which is created by amalgamating computer science and artificial intelligence. Using NLP, we are concerned with interactions between computers and human natural languages. Then the BRAIN check whether it is a question or an action, if it is a action than the action is performed by the voice assistant and acknowledgment is given to the user via a synthesis voice or if it is a question than it is search in dialog box or knowledge base and then response via a synthesis voice to the user. Our Voice assistant uses google text-to-speech API to understand all the words spoken by the user, and based on certain conditions that satisfy being a command the voice assistant sends responses to the user.

# **CLASS DIAGRAM:**

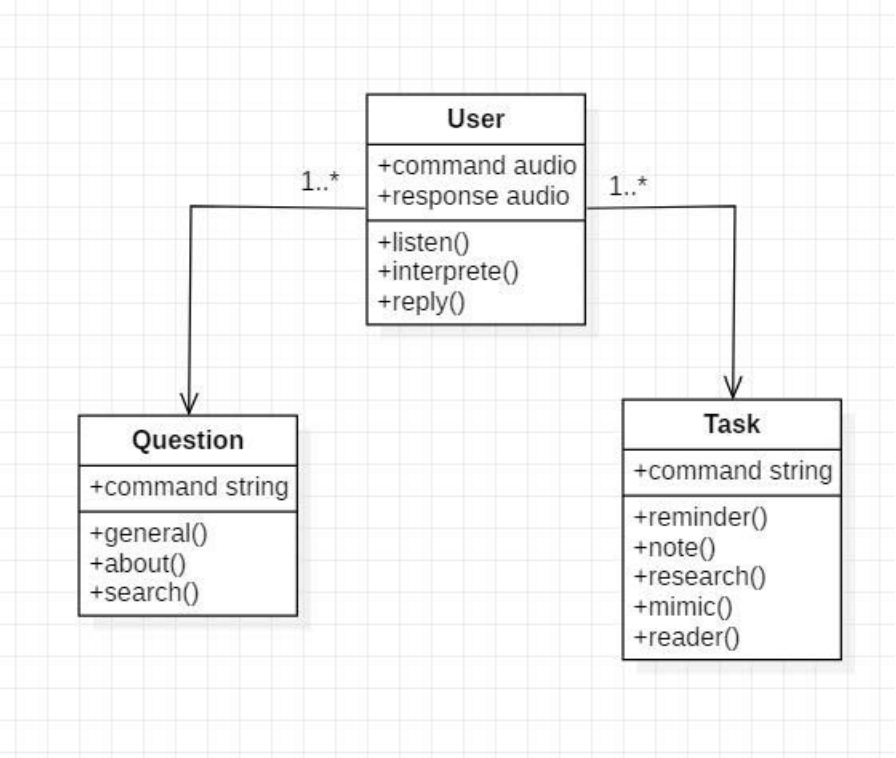


Figure 6:- class Diagram of this AI Personal Assistant

# **USE-CASE DIAGRAM:**

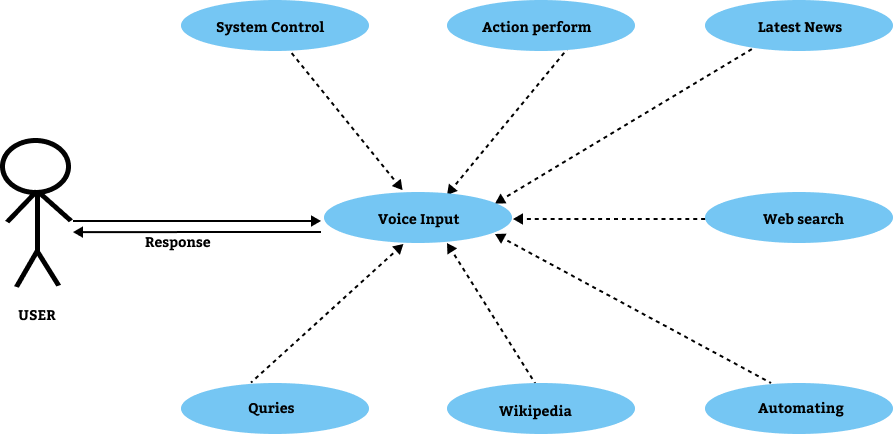


Figure 7:- Use case diagram

# **ACTIVITY DIAGRAM:**

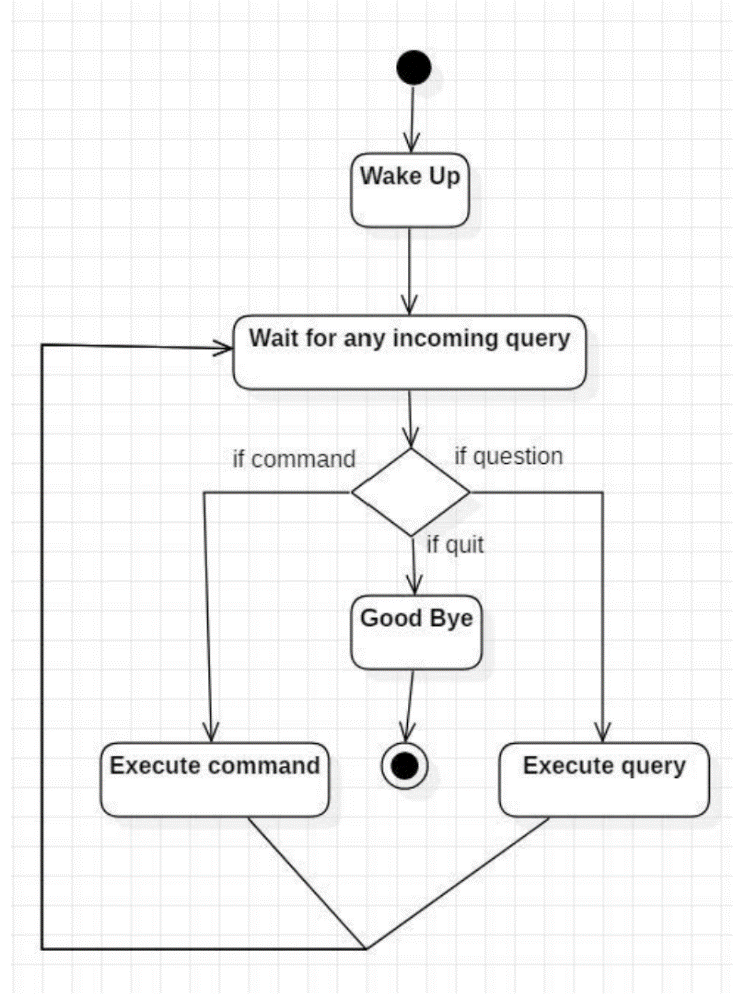


Figure 8:- Activity Diagram of this AI Personal Assistant

# **DATAFLOW DIAGRAM:**

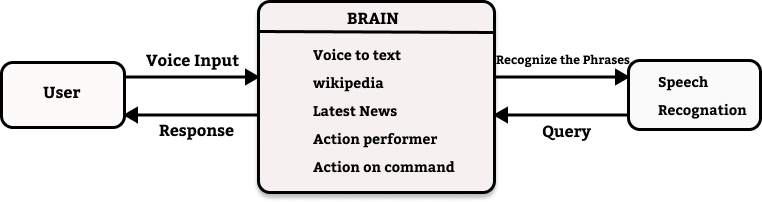
****

Figure 9:- Data flow Diagram of this AI Personal Assistant

A data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data stores, and the various subprocesses the data moves through. DFDs are built using standardized symbols and notation to describe various entities and their relationships.

Data flow diagrams visually represent systems and processes that would be hard to describe in just words. You can use these diagrams to map out an existing system and make it better or to plan out a new system for implementation. Visualizing each element makes it easy to identify inefficiencies and produce the best possible system. Physical and logical data flow diagrams. Before actually creating your data flow diagram, you’ll need to determine whether a physical or logical DFD best suits your needs. If you’re new to data flow diagrams, don’t worry—the distinction is pretty straightforward.

The user gives the input in the form of voice; this voice command is recognized by the application. Then it will check whether it is the authorized user, then action is performed as per the command given by the user. Command given is compared as a form of action and question and responded with the dialog box or search through the knowledge base.

Input is given by user in the form of voice. Google Voice API will convert this voice data in text form and then the action is performed by the voice assistant according to the command given by the user by comparing with the dialog box and knowledge base.



# **SEQUENCE DIAGRAM:**

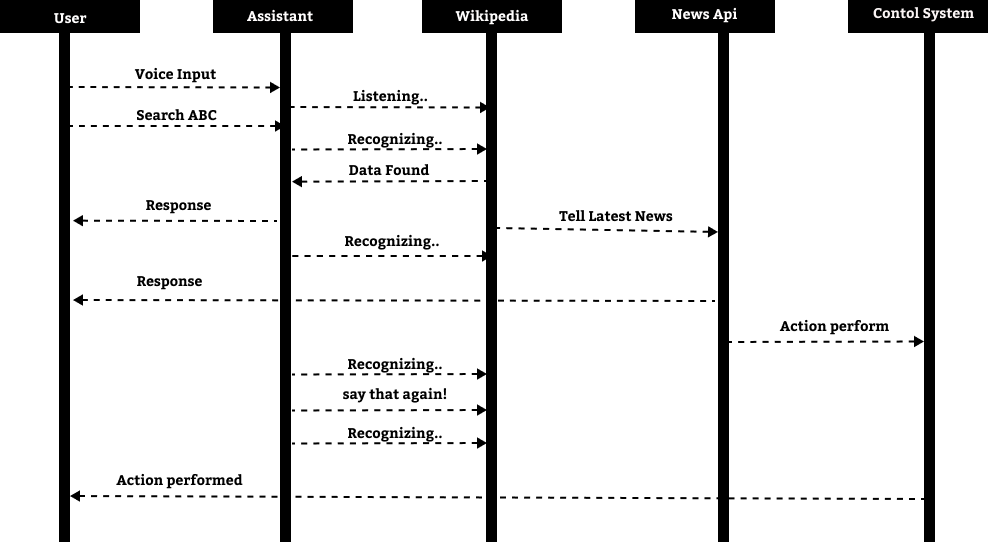
****

Figure 10:- Sequential Diagram

# **COMPONENT DIAGRAM:**

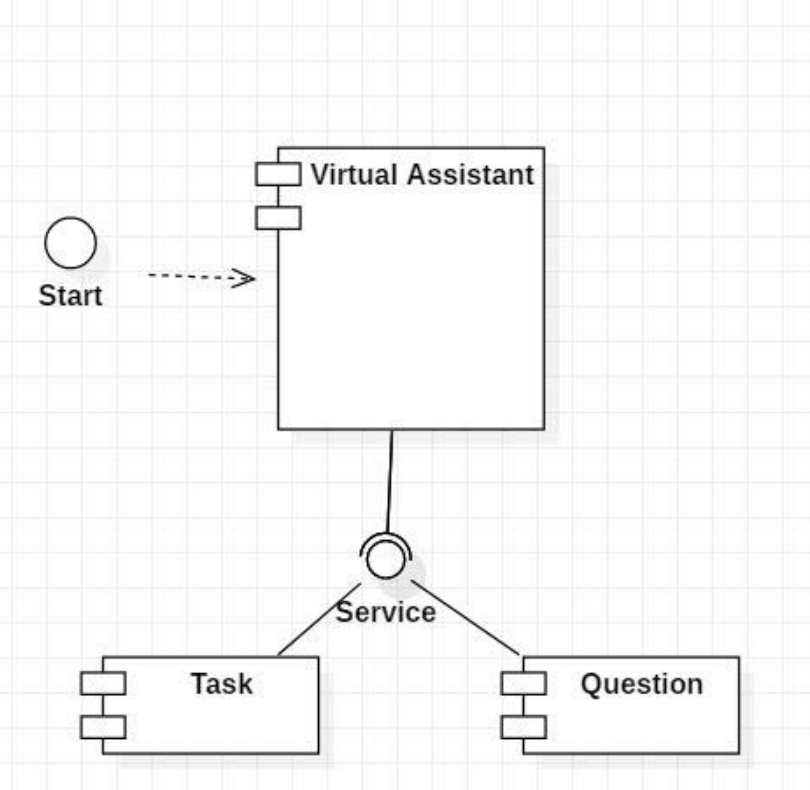
****

Figure 11:- component Diagram

# **DEPLOYMENT DIAGRAM:**

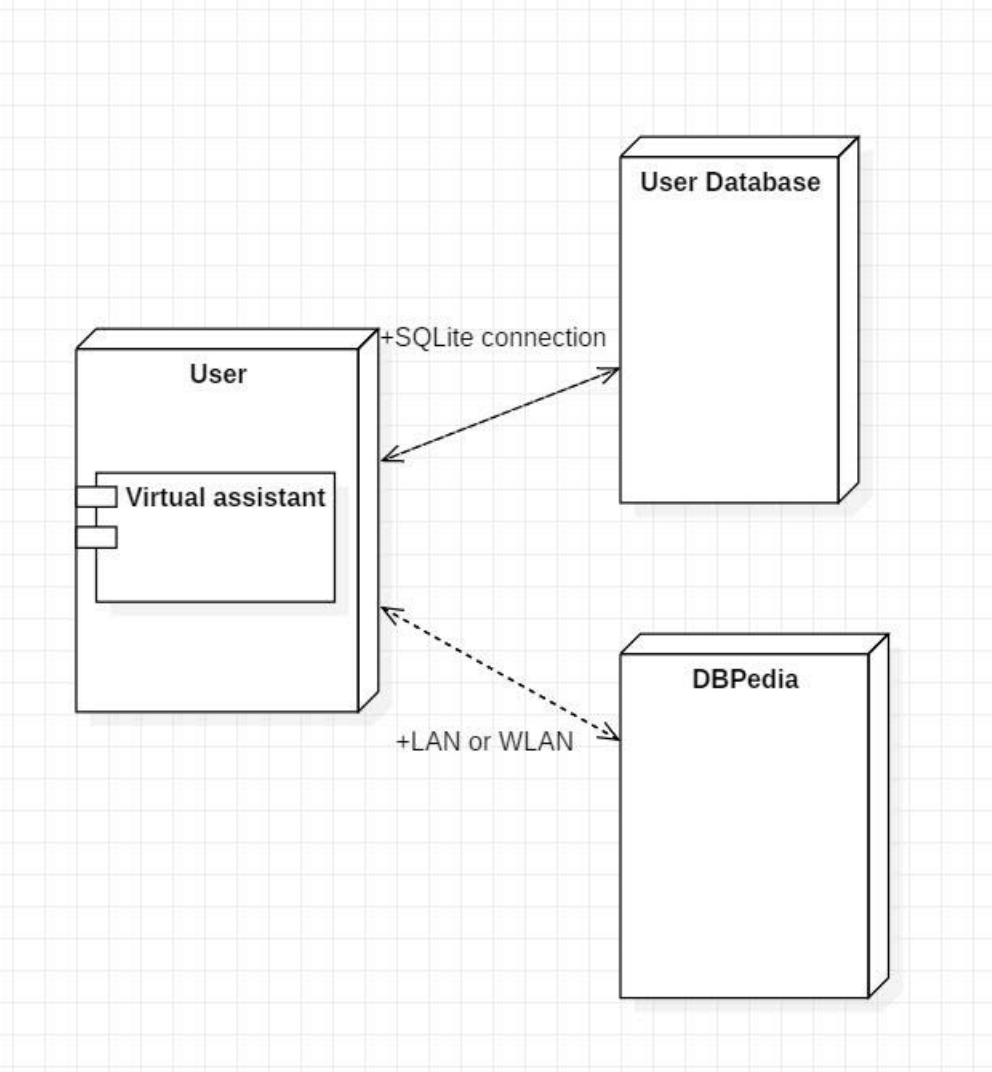


Figure 12:- Deployment Diagram

The above sequence diagram shows how an answer asked by the user is being fetched from internet. The audio query is interpreted and sent to Web scraper. The web scraper searches and finds the answer. It is then sent back to speaker, where it speaks the answer to user.

The user sends command to virtual assistant in audio form. The command is passed to the interpreter. It identifies what the user has asked and directs it to task executer. If the task is missing some info, the virtual assistant asks user back about it. The received information is sent back to task and it is accomplished. After execution feedback is sent back to user.

**SOFTWARE DEVELOPMENT LIFE CYCLE**

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software’s. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

* SDLC is the acronym of Software Development Life Cycle.
* It is also called as Software Development Process.
* SDLC is a framework defining tasks performed at each step in the software development process.
* ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

## **What is SDLC?**

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.

A typical Software Development Life Cycle consists of the following stages −

### **Stage 1: Planning and Requirement Analysis**

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

### **Stage 2: Defining Requirements**

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

### **Stage 3: Designing the Product Architecture**

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

### **Stage 4: Building or Developing the Product**

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

### **Stage 5: Testing the Product**

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

### **Stage 6: Deployment in the Market and Maintenance**

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

# **SDLC Models:**

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Following are the most important and popular SDLC models followed in the industry −

* Waterfall Model

# **PHASES OF SDLC:**

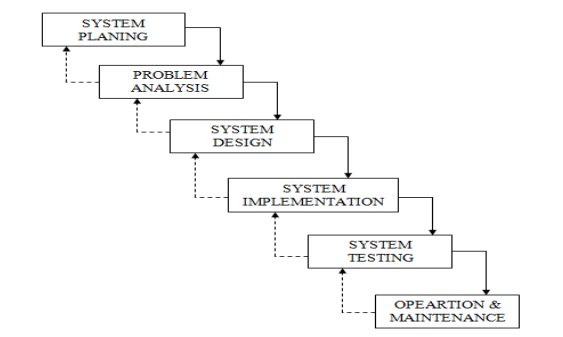


Figure 13:- Waterfall Diagram...

* **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* **System Design −** The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* **Integration and Testing −** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* **Deployment of system −** Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* **Maintenance −** There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

**IMPLEMENTATION**

# **SAMPLE CODE:**

# **(Alpha.py)**

#A project on AI personal assistant

from \_\_future\_\_ import with\_statement

import pyttsx3

import speech\_recognition as sr

import datetime

import wikipedia

import webbrowser

import os

import random

import cv2

import pywhatkit as kit

import sys

import pyautogui

import time

import operator

import requests

import wolframalpha

import json

import winshell

engine = pyttsx3.init('sapi5')

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[0].id)

engine.setProperty('rate', 180)

def speak(audio):

    engine.say(audio)

    engine.runAndWait()

def wishMe():

    hour = int(datetime.datetime.now().hour)

    if hour>=0 and hour<12:

        speak("Good Morning!")

        print("Good Morning!")

    elif hour>=12 and hour<18:

        speak("Good Afternoon!")

        print("Good Afternoon!")

    else:

        speak("Good Evening!")

        print("Good Evening!")

    speak("I am Alpha your personal Assistant and study buddy")

    print("I am Alpha your personal Assistant and study buddy")

def takeCommand():

    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')

        print(f"User said: {query}\n")

    except Exception as e:

        print("Say that again please...")

        return "None"

    return query

if \_\_name\_\_ == "\_\_main\_\_":

    wishMe()

    while True:

        speak("Tell me how can I help you now?")

        statement = takeCommand().lower()

        if statement==0:

            continue

        query = takeCommand().lower()

        if "good bye" in query or "ok bye" in query or "stop" in query:

            speak('your personal assistant Alpha is shutting down,Good bye')

            print('your personal assistant Alpha is shutting down,Good bye')

            break

        elif 'time' in query:

            strTime=datetime.datetime.now().strftime("%H:%M:%S")

            speak(f"the time is {strTime}")

        if 'open chrome' in query:

            os.startfile('C:\Program Files\Google\Chrome\Application\chrome.exe')

        if 'wikipedia' in query:

            speak('Searching Wikipedia...')

            query = query.replace("wikipedia", "")

            results = wikipedia.summary(query, sentences=2)

            speak("According to Wikipedia")

            print(results)

            speak(results)

        elif 'search on youtube' in query:

            query = query.replace("search on youtube", "")

            webbrowser.open(f"www.youtube.com/results?search\_query={query}")

        elif 'open youtube' in query:

            speak("what you will like to watch ?")

            qrry = takeCommand().lower()

            kit.playonyt(f"{qrry}")

        elif 'close chrome' in query:

            os.system("taskkill /f /im chrome.exe")

        elif 'close youtube' in query:

            os.system("taskkill /f /im msedge.exe")

        elif 'open google' in query:

            speak("what should I search ?")

            qry = takeCommand().lower()

            webbrowser.open(f"{qry}")

            results = wikipedia.summary(qry, sentences=2)

            speak(results)

        elif 'close google' in query:

            os.system("taskkill /f /im msedge.exe")

        elif 'play music' in query:

            music\_dir = 'E:\Musics'

            songs = os.listdir(music\_dir)

            os.startfile(os.path.join(music\_dir, random.choice(songs)))

        elif 'play iron man movie' in query:

            npath = "C:\\Users\\piyus\\Music"

            os.startfile(npath)

        elif 'close movie' in query:

            os.system("taskkill /f /im vlc.exe")

        elif 'close music' in query:

            os.system("taskkill /f /im vlc.exe")

        elif 'the time' in query:

            strTime = datetime.datetime.now().strftime("%H:%M:%S")

            speak(f"Sir, the time is {strTime}")

        elif "shut down the system" in query:

            os.system("shutdown /s /t 5")

        elif "restart the system" in query:

            os.system("shutdown /r /t 5")

        elif "Lock the system" in query:

            os.system("rundll32.exe powrprof.dll,SetSuspendState 0,1,0")

        elif "open notepad" in query:

            npath = "C:\WINDOWS\system32\\notepad.exe"

            os.startfile(npath)

        elif "close notepad" in query:

            os.system("taskkill /f /im notepad.exe")

        elif "open command prompt" in query:

            os.system("start cmd")

        elif "close command prompt" in query:

            os.system("taskkill /f /im cmd.exe")

        elif "open camera" in query:

            cap = cv2.VideoCapture(0)

            while True:

                ret, img = cap.read()

                cv2.imshow('webcam', img)

                k = cv2.waitKey(50)

                if k==27:

                    break;

            cap.release()

            cv2.destroyAllWndows()

        elif "go to sleep" in query:

            speak(' alright then, I am switching off')

            sys.exit()

        elif "take screenshot" in query:

            speak('tell me a name for the file')

            name = takeCommand().lower()

            time.sleep(3)

            img = pyautogui.screenshot()

            img.save(f"{name}.png")

            speak("screenshot saved")

        elif "what is my ip address" in query:

            speak("Checking")

            try:

                ipAdd = requests.get('https://api.ipify.org').text

                print(ipAdd)

                speak("your ip adress is")

                speak(ipAdd)

            except Exception as e:

                speak("network is weak, please try again some time later")

        elif "volume up" in query:

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

            pyautogui.press("volumeup")

        elif "volume down" in query:

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

            pyautogui.press("volumedown")

        elif "mute" in query:

            pyautogui.press("volumemute")

        elif "refresh" in query:

            pyautogui.moveTo(1551,551, 2)

            pyautogui.click(x=1551, y=551, clicks=1, interval=0, button='right')

            pyautogui.moveTo(1620,667, 1)

            pyautogui.click(x=1620, y=667, clicks=1, interval=0, button='left')

        elif "scroll down" in query:

            pyautogui.scroll(1000)

        elif "drag visual studio to the right" in query:

            pyautogui.moveTo(46, 31, 2)

            pyautogui.dragRel(1857, 31, 2)

        elif "rectangular spiral" in query:

            pyautogui.hotkey('win')

            time.sleep(1)

            pyautogui.write('paint')

            time.sleep(1)

            pyautogui.press('enter')

            pyautogui.moveTo(100, 193, 1)

            pyautogui.rightClick

            pyautogui.click()

            distance = 300

            while distance > 0:

                pyautogui.dragRel(distance, 0, 0.1, button="left")

                distance = distance - 10

                pyautogui.dragRel(0, distance, 0.1, button="left")

                pyautogui.dragRel(-distance, 0, 0.1, button="left")

                distance = distance - 10

                pyautogui.dragRel(0, -distance, 0.1, button="left")

        elif "close paint" in query:

                os.system("taskkill /f /im mspaint.exe")

        elif "who made you" in query or "who created you" in query or "Name the buddy created you" in query or "Name the persons created you" in query:

            print('I was created by Aman piyush, Nikitha Bhavani, and Kalyan sumanth in python language in visual studio.')

            speak('I was created by Aman piyush, Nikitha Bhavani, and Kalyan sumanth in python language in visual studio')

        elif 'who are you' in query or 'what can you do' in query:

            speak('I am Alpha your personal assistant. I am programmed to minor tasks like'

                'opening youtube,google chrome, gmail and stackoverflow ,predict time,take a photo,search wikipedia,predict weather'

                'In different cities, get top headline news from times of india and you can ask me computational or geographical questions too!')

        elif "open notepad and write my channel name" in query:

            pyautogui.hotkey('win')

            time.sleep(1)

            pyautogui.write('notepad')

            time.sleep(1)

            pyautogui.press('enter')

            time.sleep(1)

            pyautogui.write("How To Manual", interval = 0.1)

        elif 'type' in query: #10

            query = query.replace("type", "")

            pyautogui.write(f"{query}")

        elif 'maximize this window' in query:

            pyautogui.hotkey('alt', 'space')

            time.sleep(1)

            pyautogui.press('x')

        elif 'google search' in query:

            query = query.replace("google search", "")

            pyautogui.hotkey('alt', 'd')

            pyautogui.write(f"{query}", 0.1)

            pyautogui.press('enter')

        elif 'youtube search' in query:

            query = query.replace("youtube search", "")

            pyautogui.hotkey('alt', 'd')

            time.sleep(1)

            pyautogui.press('tab')

            pyautogui.press('tab')

            pyautogui.press('tab')

            pyautogui.press('tab')

            time.sleep(1)

            pyautogui.write(f"{query}", 0.1)

            pyautogui.press('enter')

        elif 'open new window' in query:

            pyautogui.hotkey('ctrl', 'n')

        elif 'open private window' in query:

            pyautogui.hotkey('ctrl', 'shift', 'n')

        elif 'minimise this window' in query:

            pyautogui.hotkey('alt', 'space')

            time.sleep(1)

            pyautogui.press('n')

        elif 'open history' in query:

            pyautogui.hotkey('ctrl', 'h')

        elif 'open downloads' in query:

            pyautogui.hotkey('ctrl', 'j')

        elif 'previous tab' in query:

            pyautogui.hotkey('ctrl', 'shift', 'tab')

        elif 'next tab' in query:

            pyautogui.hotkey('ctrl', 'tab')

        elif 'close tab' in query:

            pyautogui.hotkey('ctrl', 'w')

        elif 'close window' in query:

            pyautogui.hotkey('ctrl', 'shift', 'w')

        elif 'clear browsing history' in query:

            pyautogui.hotkey('ctrl', 'shift', 'delete')

        elif 'close chrome' in query:

            os.system("taskkill /f /im chrome.exe")

        elif "weather" in query:

            api\_key="6e0ac4770a4b0e232d82bdf75ad1aa6b"

            base\_url="https://api.openweathermap.org/data/2.5/weather?"

            speak("what is the city name")

            city\_name=takeCommand()

            complete\_url=base\_url+"appid="+api\_key+"&q="+city\_name

            response = requests.get(complete\_url)

            x=response.json()

            if x["cod"]!="404":

                y=x["main"]

                current\_temperature = y["temp"]

                current\_humidiy = y["humidity"]

                z = x["weather"]

                weather\_description = z[0]["description"]

                speak(" Temperature in kelvin unit is " +

                    str(current\_temperature) +

                    "\n humidity in percentage is " +

                    str(current\_humidiy) +

                    "\n description  " +

                    str(weather\_description))

                print(" Temperature in kelvin unit = " +

                    str(current\_temperature) +

                    "\n humidity (in percentage) = " +

                    str(current\_humidiy) +

                    "\n description = " +

                    str(weather\_description))

        elif "news" in query:

                url = url = ('https://newsapi.org/v2/top-headlines?'

                            'country=in&'

                            'apiKey=c5a1b15654634a259327c86876d85193')

                try:

                    response = requests.get(url)

                except:

                    speak("Please check your connection")

                news = json.loads(response.text)

                for new in news["articles"]:

                    print(str(new["title"]), "\n")

                    speak(str(new["title"]))

                    engine.runAndWait()

                    print(str(new["description"]), '\n')

                    speak(str(new["description"]))

                    engine.runAndWait()

        elif "exit" in query or "quit" in query:

            exit()

**TESTING**

# **INTRODUCTION:**

Testing is a vast domain. When a software application is developed, it goes through various stages of testing. The tests may vary from being functional to non-functional, subject to test requirements.

There are two ways for QAs to carry out tests:

* By manually executing test cases as mentioned in the written test plan
* By automating test scenarios using frameworks like Selenium

In a tech-advanced world where automation tools and frameworks are making life easier for test engineers, it would be normal to assume that manual testing is obsolete. This is, however, completely untrue. This article will explain everything one should know about manual testing. It will also help readers understand why manual testing can’t be avoided altogether.

Manual testing, as the term suggests, refers to a test process in which a QA manually tests the software application in order to identify bugs. To do so, QAs follow a written test plan that describes a set of unique test scenarios. The QA is required to analyse the performance of the web or mobile application from an end user’s perspective.

QAs verify the actual behaviour of software against expected behaviour, and any difference is reported as a bug.

Let’s take a simple example to explain this. A developer has created a website and wants to test it for functionality. In this case, the expected behaviour is that the user must be able to enter the username and password and submit the credentials by clicking the Login button.

However, when the test is executed, the Login button doesn’t redirect the user to the home page. In such a case, the QA will report a bug to the developer. Manual testing is an essential part of any test strategy as it helps QAs gain deeper insight from an end user’s perspective. Since manual testing is carried out by a human without the intervention of test automation frameworks, it judges software from the most important metric: User Experience.

Manual testing plays a vital role in exploratory testing or in test cases that are executed once or twice. This helps QAs to discover bugs in the early stages of the development cycle.

## **Types of Manual Testing**

## **1. White Box Testing**

White box Testing, also known as glass box or transparent testing, is an approach in which the QA is familiar with the internal code or structure of the application. It is primarily used for unit testing. White box Testing also covers specific techniques like data flow testing, control flow testing, decision coverage, and path testing, and a few others.

## **2. Black Box Testing**

Black-box testing is a test approach in which the QA doesn’t have any knowledge about the underlying code or structure of the application. The QA interacts with the software application just like an end-user to test its functional and non-functional behaviour. This helps to discover some bugs typically overlooked in the earlier stages.

## **3. Grey Box Testing**

Grey-Box test approach is the combination of both white box and black box testing techniques. The main aim of this approach is to identify any bugs present either due to inappropriate usage or any structural flaws.

# **How to perform Manual Testing**

Here’s how to perform manual testing step by step:

* Analyse requirements from the software requirement specification document
* Create a clear test plan
* Write test cases that cover all the requirements defined in the document
* Get test cases reviewed by the QA lead
* Execute test cases and detect any bugs
* Report bugs, if any, and once fixed, run the failed tests again to re-verify the fixes.

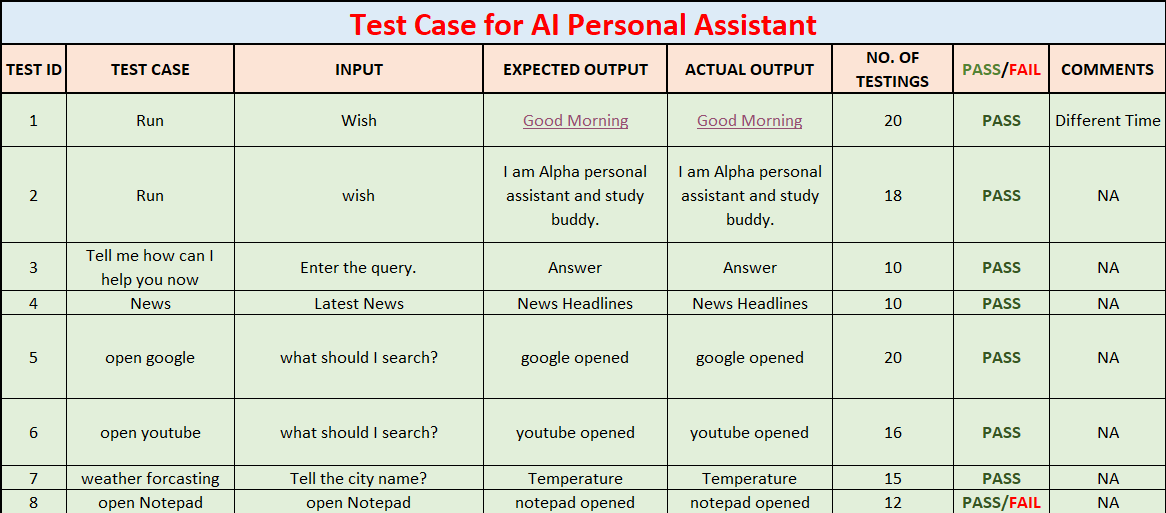


Figure 14:- Test cases of this AI personal Assistant

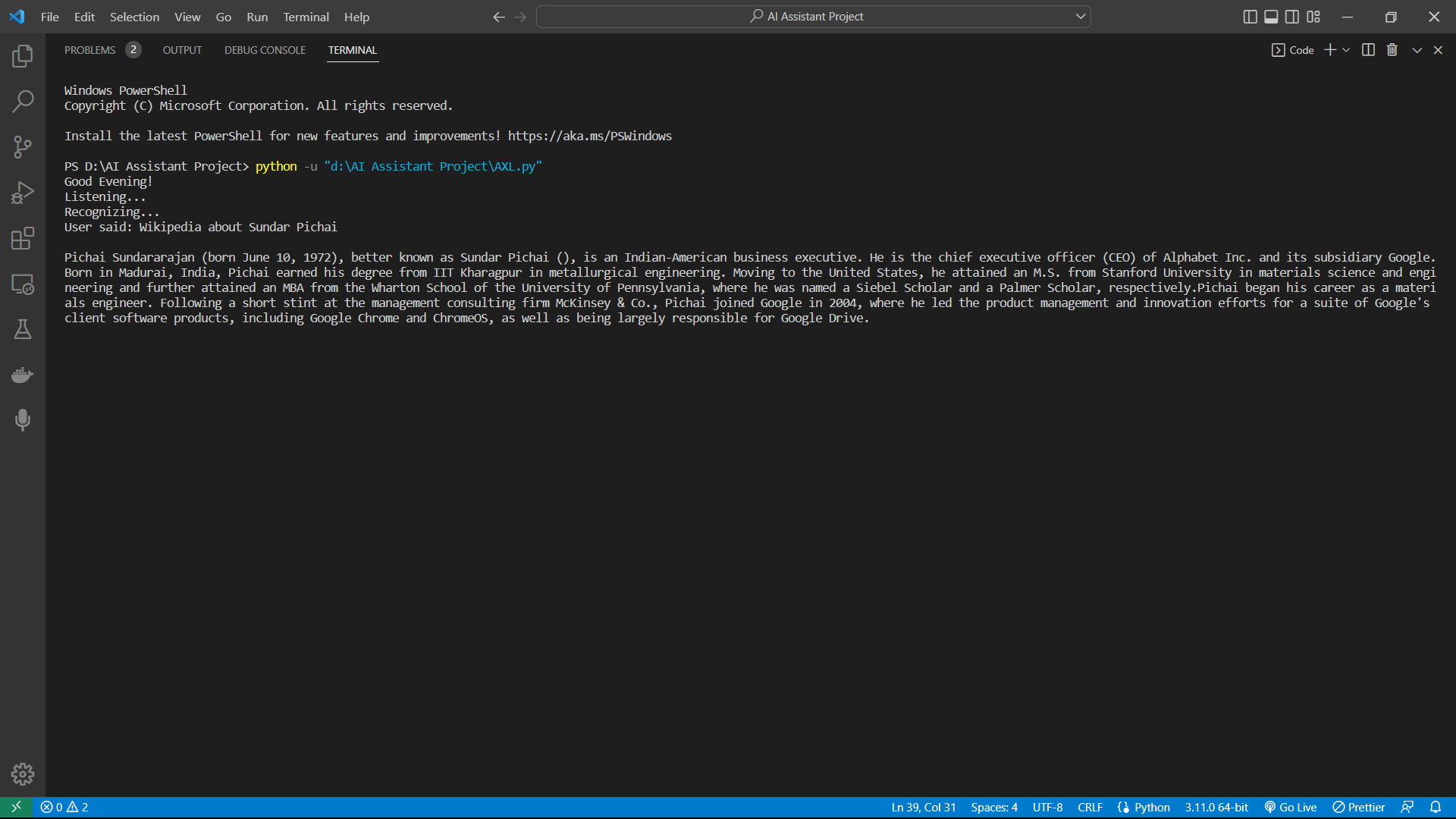


Figure 15:- Output

Although the approach towards software development is continuously demanding a shift towards test automation, manual testing will always be essential. In fact, manual testing is a good way to assess aesthetic aspects relevant to end-users such as how well the web elements render, how easy it is to navigate through an application, and so on.

**OUTPUT SCREEN**

# **SCREENSHOTS:**

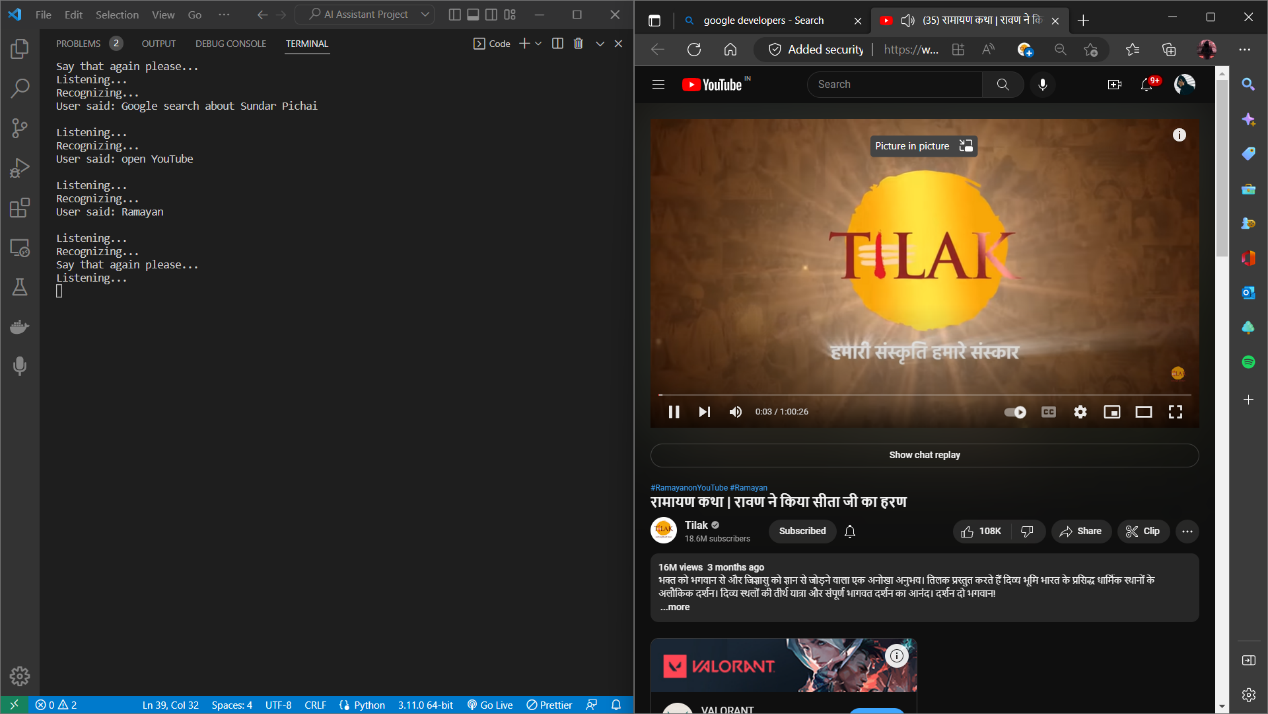


Figure 16:- Output

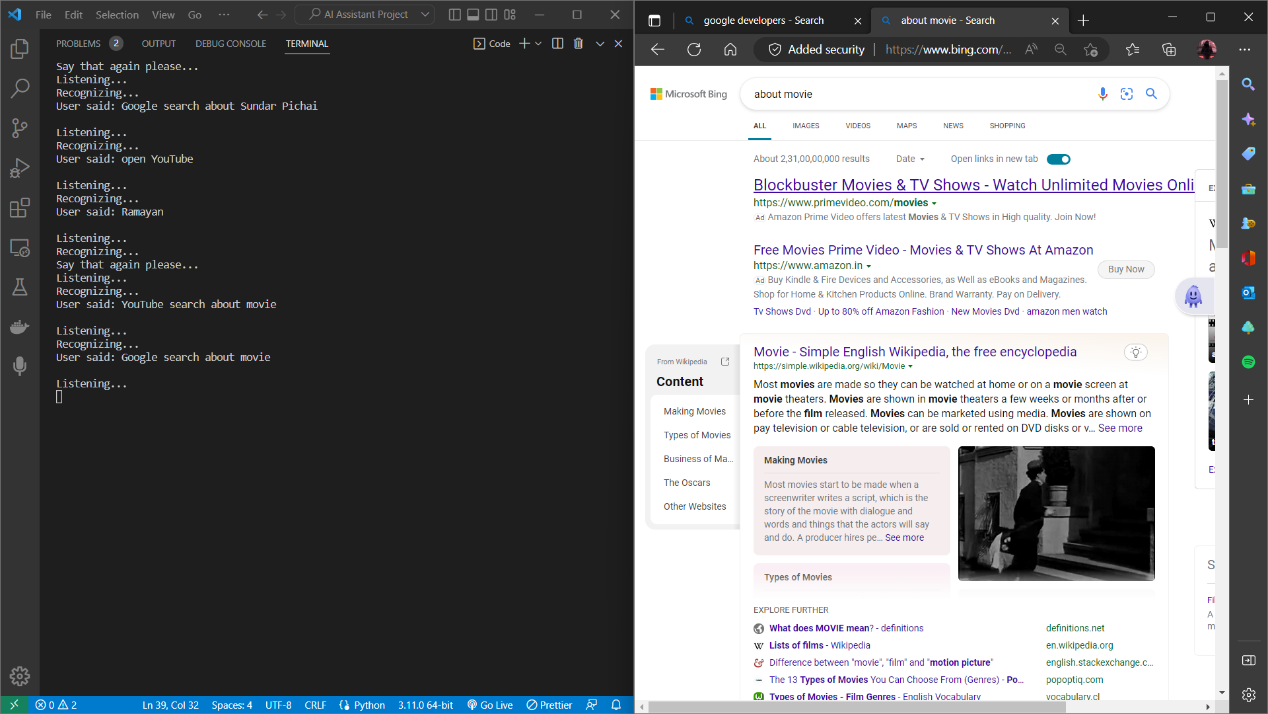


Figure 17:- Output

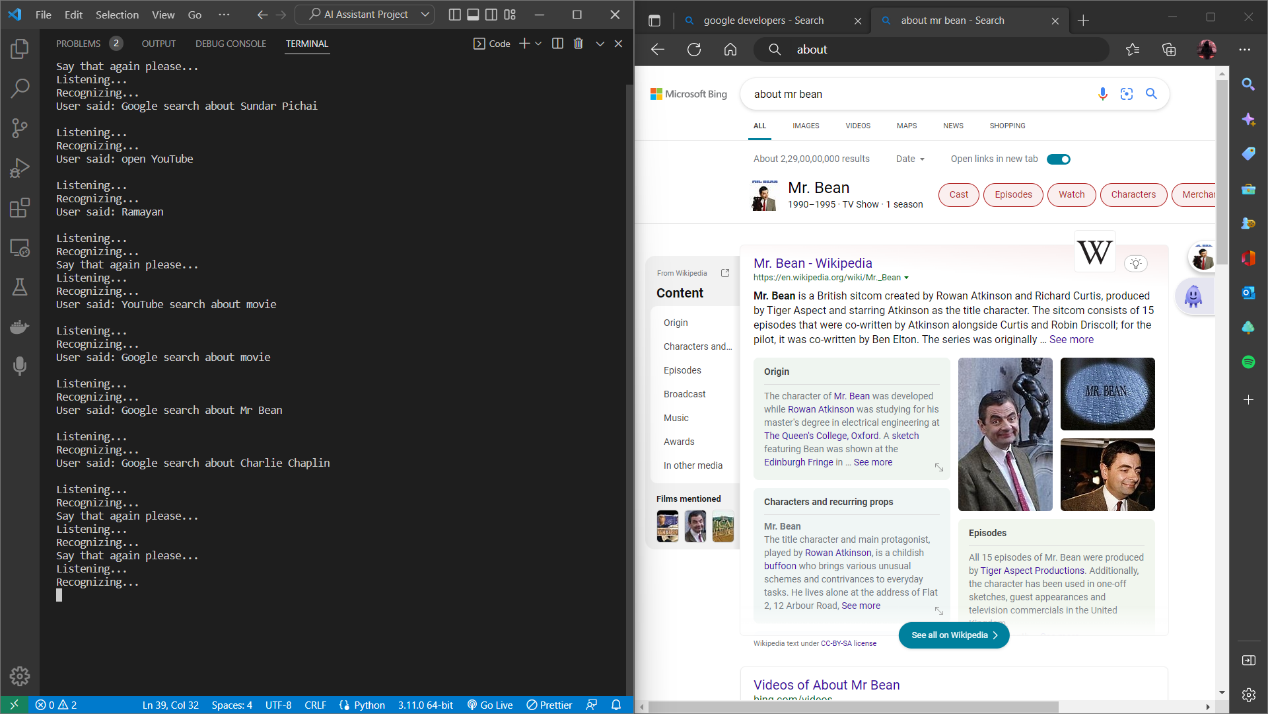


Figure 18:- Output

When assistant start running it will start listening if you command the assistant.

User: Open YouTube

Assistant: it will ask what do you like to watch?

User: I like to watch Ramayan.

Assistant: It will again ask how may I help you?

User: Open Google

Assistant: What do you like to search?

User: About Charli chaplin.

**CONCLUSION AND FUTURE SCOPE**

# **CONCLUSION:**

The main objective of the project is to develop a AI personal Assistant Application and had taken a wide range of literature review in order to achieve all the tasks, where I came to know about this application is tested in real-time and can be used effectively 24/7 with flawless features

This system is designed in such a method where in the user can accommodate to it effortlessly. Our proposed system BRAIN – The A.I. a personal voice assistant can be implemented using the face recognition and using speech recognition module thus makes the system more secure and robust. The contributions of Smart Voice Assistant are twofold. First, the face recognition technique make it more secure and robust to use, Secondly, it is the voice control application that provides enhancements to all applications running on a system by synthesizing commands set from onscreen context.

# **FUTURE SCOPE:**

Using this system as a framework, the system can be expanded to features security. Security is important these days so it can be combined with this system to give more advanced security features. In this, the voice authentication technology can be implemented for more security. More advancement are possible like operating on various tones or accents from different regions that mean it should be able to perform operations on various voice tones and accents. Further modifications are possible like learning the answer of questions that are not known by the voice assistant and replying whenever next time the same question is put up by the user.

**REFERENCES**

# **Websites Referred:**

* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.pythonprogramming.net](http://www.pythonprogramming.net)
* [www.tutorialspoint.com](http://www.tutorialspoint.com)
* [www.codeacademy.com](http://www.codeacademy.com)
* [www.medium.com](http://www.medium.com)
* [www.google.co.in](http://www.google.co.in)
* [www.geekforgeek.org](http://www.geekforgeek.org)

# **Websites Referred:**

* Python programming
* Learning Python
* AL with Python

# **YouTube Referred:**

* Code with Harry
* Apni Kaksha
* CS Dojo
* Edureka!

# **Blog or Documentation Referred:**

* <https://medium.com/codex/making-your-own-ai-virtual-assistant-with-python-5c2046dadfa7>
* <https://medium.com/analytics-vidhya/a-guide-to-your-own-a-i-voice-assistant-using-python-17f79c94704>
* <https://medium.datadriveninvestor.com/7-simple-steps-to-building-a-virtual-assistant-with-python-eeb5ad371fff>
* <https://medium.com/analytics-vidhya/privacy-virtual-assistant-powered-by-machine-learning-in-python-32578e0d56be>
* <https://dlabs.ai/blog/top-ai-blogs-and-websites-to-follow/>