**Minor Project Report**

**ON**

**DESKTOP VOICE ASSISTANT**

A Minor Project Report Submitted

in Partial Fulfillment of the Requirements

for the degree of **Masters of Computer Application**



|  |  |
| --- | --- |
|  |  |

**Under Guidance of Approved by Asst. Prof. Brijesh K. Soni Dr. Akhilesh A. Waoo**

**Submitted by**

**Ankit Singh**

**(B2192R10700036)**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**A.K.S UNIVERSITY,**

**SHERGANJ SATNA , (M.P)**



**ACKNOWLEDGEMENT**

A project owes its success from commencement to completion, to the people in love with creativity at various stages. Let us in this page express my gratitude to all those who helped us in various stage of this study.

We wish to express our sincere gratitude indebtedness to **Mr.BRIJESH K. SONI** (Department of Computer Science Engineering, AKSU) for introducing the present topic and for his inspiring guidance, constructive criticism and valuable suggestion throughout this project work.

We are also thankful to our Parents for their true help and inspiration. Last but not least, we pay our sincere thanks and gratitude to all the staff members and our project partner at **AKS UNIVERSITY** for their support and for making our training valuable and fruitful.

Date:

Place:

****

**DECLARATION**

I hereby declare that the project work on **“Desktop Voice Assistant”** submitted to the **AKS UNIVERSITY**, is a record of original work done by us under the guidance of **Mr. Brijesh K. Soni**, (Department of Computer Science Engineering, AKSU), **AKS University**, Satna and this report is submitted in the fulfillment of the requirements for the award of the degree of Master of Computer Application.

Signature: Signature:

(Guide) (Student)

Date:



**CERTIFICATE**

It is certified that the work contained in the project report titled **“Desktop Voice Assistant”**, by Ankit Singh has been carried out under my supervision and that this work has not been submitted elsewhere for any other degree.

**Dr. Akhilesh A. Wao**

Head of Department

Department of Computer Science and Engineering

AKS University, Satna

**Mr. Brijesh K. Soni**

Assistant Professor

Department of Computer Science and Engineering

AKS University, Satna

**CONTENTS**

1. **Introduction**
2. **Project Overview And Summary**
   1. Literature Survey
   2. Overview Of Project
   3. Future Scope
3. **System Specification**
   1. Hardware Requirements
   2. Software Requirements
4. **Technology Used**
   1. Frontend
   2. Backend
5. **Software Architecture**
   1. Software Design
   2. Block Diagram
   3. Flow Chart
6. **Project Description**
   1. Used Python Libraries
   2. Project Modules
   3. Database Design
7. **Features And Security**
   1. Output Screenshots
8. **Implementation Of Code**
9. **References**
10. **Conclusion**
11. **INTRODUCTION**

The developments of artificial intelligence (AI) systems which can make virtual human are gaining popularity. Human does not learns to talk with a device, but it’s the machine that learns to speak with a humans. Virtual assistants are software program that help you to easily do your day to day works like displaying climate reviews, on off devices and so on. They take input as command through text or through voice. Voice based assistants need a wake phrase to start, followed by the command. There are many voice assistance like Apple’s Siri, Amazon’s Alexa and Microsoft’s Cortana. Such as stream movie on YouTube, Search on Wikipedia and Google, Open websites on web browser, Open applications. the voice assistant we've got advanced is a computer-primarily based built using python modules and libraries. This assistant is just a simple model that might carry out all the simple responsibilities which have been mentioned above however cutting-edge generation is even though right in its miles nevertheless to be merged with Machine Learning and Internet Of Things (IoT) for better improvements. Personal assistants software program improves person productiveness by means of dealing with routine obligations of the consumer and by using providing records from an internet supply to the consumer. So with a virtual assistant, we are able to be capable of manage many stuff in our desktop/laptop around us with simplest one platform.

Human interaction is rapidly being supplanted by. Performance is one of the key reasons behind this shift. Rather than progress, technology has undergone a significant transformation. In today's world, we use technologies like Machine Learning and Neural Networks to teach our machines to do their jobs on their own or to think like people. With the help of virtual assistants, we may now communicate with our machines in the modern world. Virtual assistants are beneficial to the elderly, the visually and physically challenged, children, and others by ensuring that interacting with machines is no longer a challenge. Even blind persons who are unable to see the computer can communicate with it just through their voice.

Our voice assistant is a desktop-based application created with Python modules and libraries. This assistant is only a basic version that can do all of the duties listed above, but current technology, while impressive, still has to be combined with Machine Learning and the Internet of Things (IoT) for greater results. The understanding and executing commands are still to reach a new level like the virtual assistant of the iron man named Jarvis. Although this is a work of fiction, it is possible to achieve with virtual assistants. All you have to do is give the assistant a command, and the assistant will take care of the rest. There will be no need to write extensive codes to execute a task with the help of voice-activated virtual assistants; the system will do it for us.

1. **PROJECT OVERVIEW AND SUMMARY**

**2.1 LITERATURE SURVEY**

This area of digital assistants having speech reputation has visible some primary advancements or inventions. This is especially due to its call for in gadgets like smart watches or health bands, speakers, Bluetooth earphones, cellular telephones, computer or desktop, TV, and so forth. Almost all the digital gadgets which are coming nowadays with voice assistants, which assist to control the device with speech recognition. A new set of strategies is being evolved constantly to improve the performance of voice computerized seek. With using voice assistants, we are able to automate the project without difficulty, simply give the center to the machine within the speech shape and all the duties might be accomplished by means of it from changing your speech into textual content shape to putting off keywords from that text and execute the question to provide outcomes to the person. This has been one of the most beneficial improvements in era.

**2.1 Overview Of Project**

The project aims to develop a **Voice Assistant** for windows based operating system. Jarvis draws its inspiration from virtual assistants like Cortana for Windows, and Siri for iOS. It has been designed to perform variety of tasks by employing certain well-defined commands. Users can interact with the assistant through voice commands. As a personal assistant, Jarvis assists the end-user with day-to-day activities like general user conversation, searching queries in google, searching for videos, live weather conditions, word meanings, **Whatsapp Automation**, basic computation etc. also it has a security system that is **Face Recognition System** which only allows the authenticated person to access it with his voice commands.

**Keywords:-**

* **Voice Assistant**
* **Automation**
* **Text-Speech-Text**
* **Voice Detection**
* **Face Recognition**
* **Python.**

**2.3 FUTURE SCOPE**

The virtual assistants who might be presently available are speedy and responsive however we nevertheless must go an extended manner. The assistants available these days are still no longer reliable in critical eventualities. The destiny of these assistants may have the virtual assistants integrated with Artificial Intelligence which incorporates Machine Learning, Neural Networks and IoT. With this technology, we are able to gain new heights. What the digital assistants can attain is lots past what we have done till now. Most of us have seen J**arvis**, that is a AI Based personal voice assistant developed by **Tony Stark** in **Iron Man** movie which is even though fictional but this has set new requirements of what we will achieve using voice activated digital assistant.

This is the first step in the new era of automation in future almost every task can be done through the voice command apart from desktop and devices assistant can also automates home appliance such as TV, AC, Lights, Fan etc.

1. **SYSTEM SPECIFICATION**

**3.1 HARDWARE REQUIREMENTS**

* **Intel Core i3 Or Above.**
* **2 GB Ram**
* **512 KB Cache Memory**
* **10 GB (Free Space) HDD**
* **Microphone**
* **Webcam**

**3.2 SOFTWARE REQUIREMENTS**

* Windows 8 And Above
* Python 3.10.6
* SQLite database
* OpenWeatherMap API
* WolFramAlpha API

**4. TECHNOLOGY USED**

**4.1 Frontend**

1. **Python**

**Introduction Of Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

Created by Guido van Rossum in 1991.

Its high-level built in data structure, 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.

 It is used for:

* Web Development (server-side),
* Software Development,
* Mathematics,
* System Scripting,
* Data Science,
* Machine Learning.

### **Why Python?**

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-oriented way or a functional way.

### **What can Python do?**

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

**Advantages Of Python**

* Versatile, Easy to read, learn and write
* Open source and large active community base
* Extensive support libraries(NumPy for numerical calculations, Pandas for data analytics, etc.)
* Presence of third-party modules.

**2.Visual Studio Code**

**Introduction Of VS Code**

Visual Studio Code, also commonly referred to as VS Code,it is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool among 82,000 respondents, with 70% reporting that they use it.

Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

On November 18, 2015, the source of Visual Studio Code was released under the MIT License, and made available on GitHub. Extension support was also announced. On April 14, 2016, Visual Studio Code graduated from the public preview stage and was released to the Web.

**Why VS Code?**

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C#, Java, JavaScript, Go, Node.js, Python, C++, C, Rust and Fortran. It is based on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

VS Code provide more handy features to the user then any other code editor that’s why programmers and developers uses vs code aslo it has smart features like auto suggestion and user snipit etc.

**4.2 Backend**

1. **SQLite Database**

**Introduction Of SQLite**

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

## Why SQLite?

* SQLite does not require a separate server process or system to operate (serverless).
* SQLite comes with zero-configuration, which means no setup or administration needed.
* A complete SQLite database is stored in a single cross-platform disk file.
* SQLite is very small and light weight, less than 400KiB fully configured or less than 250KiB with optional features omitted.
* SQLite is self-contained, which means no external dependencies.

**SQLite With Python**

SQLite3 can be integrated with Python using **sqlite3** **module**, which was written by Gerhard Haring. It provides an SQL interface compliant with the DB-API 2.0 specification described by PEP 249. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards.

To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements.

1. **Application Programming Interface**

**Introduction Of API**

An application programming interface, or API, enables companies to open up their applications’ data and functionality to external third-party developers and business partners, or to departments within their companies. This allows services and products to communicate with each other and leverage each other’s data and functionality through a documented interface. Programmers don't need to know how an API is implemented; they simply use the interface to communicate with other products and services. API use has surged over the past decade, to the degree that many of the most popular web applications today would not be possible without APIs.

**Here’s how an API works:**

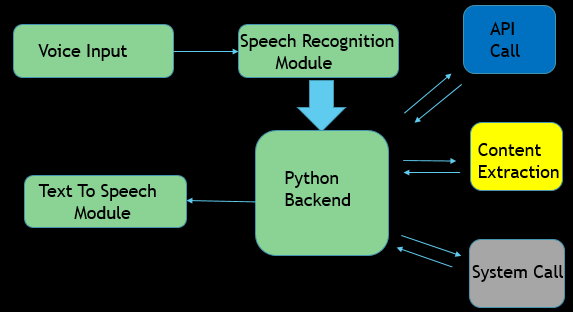
1. **A client application initiates an API call**to retrieve information—also known as a request. This request is processed from an application to the web server via the API’s Uniform Resource Identifier (URI) and includes a request verb, headers, and sometimes, a request body.
2. **After receiving a valid request**, the API makes a call to the external program or web server.
3. **The server sends a response**to the API with the requested information.
4. **The API transfers the data**to the initial requesting application.
5. **SOFTWARE ARCHITECTURE**

**5.1 Software Design**

The design consists of the following:

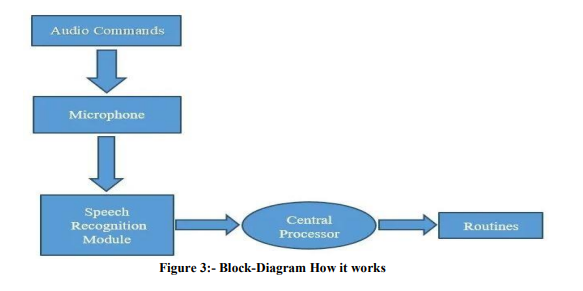
* Taking voice as a input from user.
* Conversion of the speech into text by the system.
* The converted text is then processed to get the desired output.
* The text contains some keywords that determine what queries are to be executed. If the keyword did not matched the queries in the program hence the assistant will ask the user to speak again. The output which is in the text form is converted to speech and is provided to the user. Above figure shows the flow of the voice assistant system. speech recognition is used to convert the input voice to text. This text is then sent to the processor, it detects the nature of the command and calls the related code for execution process

**Working Mechanism Of Application**



**5.2 Block Diagram**

1. User Input (Voice Command): Initially, the assistant will greet the user and then ask the user for task and then listen to it.
2. Assistant: Here the assistant consists of three sub blocks i.e.
3. Convert The Voice Input into Text:
   * Convert The Voice Input into Text: Here with the help of speech recognition module the electro will convert the input command into text.
4. Perform The Task:
   * Perform The Task: Now according to the user input the electro will perform the task.
5. Display Output:
   * Display Output: The result of the given task is shown here.

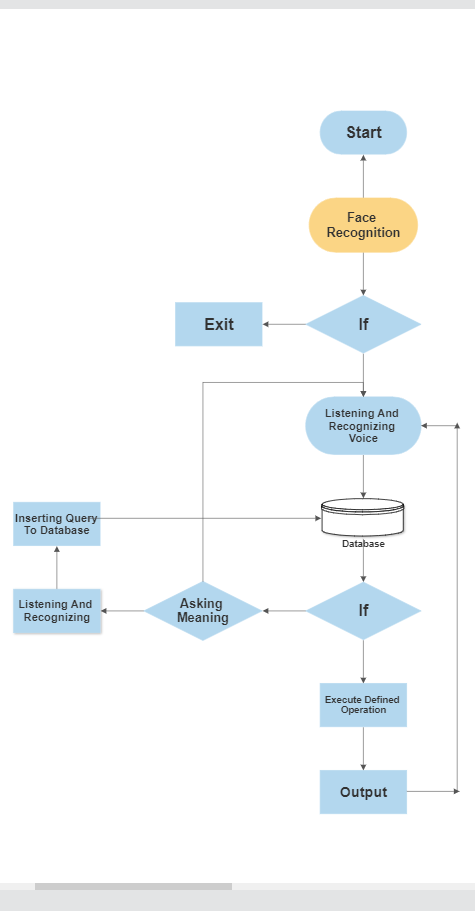
****

**5.3 Data Flow Chart**

This is the flowchart of our virtual desktop assistant.

* When we run the program, it starts with greeting the user according to time, like good morning, good afternoon.
* After this, it will start listening to the user’s voice command, and recognizes the command.
* If the given command matches any of the following command then it will search for the particular function whether it is present in the main body or not.
* Wherever it is present it will execute the command and after successful execution it will go back to the listening mode.
* If it does not match or recognize the user’s voice command then it says “Invalid input” and again go back to listening mode.
* To finish the software, the user must say turn off, and the loop will be broken.

**Data Flow Diagram**



**6. Project Description**

**6.1 Used Python Libraries**

* **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 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.

* **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.

* **face\_recognition**

Recognize and manipulate faces from Python or from the command line with the world’s simplest face recognition library. Built using dlib’s state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labled Faces In The Wild Benchmark.

* **opencv**

 OpenCV is an open source library which is supported by multiple platforms including Windows, Linux, and MacOS, and is available for use in multiple other languages as well; however, it is most commonly used in Python for Machine Learning applications, specifically in the Computer Vision domain.

* **Numpy**

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

* **sqlite3**

Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases. There is no need to install this module separately as it comes along with Python after the 2.5x version.

* **pyaudio**

PyAudio provides Python bindings for PortAudio v19, the cross-platform audio I/O library. With PyAudio, you can easily use Python to play and record audio on a variety of platforms, such as GNU/Linux, Microsoft Windows, and Apple macOS.

* **datetime**

The Date-Time module is imported to support the date and time. For example, the consumer wants to recognize the modern- day date and time or the person desires to time table a venture at a sure time. In brief this module helps instructions to manipulate date and time and carry out operations according to it handiest.

* **Webbrowser**

Web-browser module is imported to display information from web to users. If the user wants to open browser and gives input as “Open Google”. Then input is processed using this module and the Google browser is opened. The browser which is set in code will open

* **Wikipedia**

Wikipedia is an online library in python which it possible for the virtual assistant to process the queries on Wikipedia and display it to the users. This library needs an internet connection. The number of lines that the user wants to get as a result can be set manually.

* **OS module**

OS Module provides operating system dependent functionalities. If we want to perform operations of OS like data reading, data writing, or data manipulate paths then this types of functions are available in an OS module. When the these operations raise an error like “OSError” in case of any error like invalid names, paths, or arguments which may be incorrect or correct but just not accepted by the operating system.

* **Requests**

The request module allows you to send HTTP requests using Python.

The HTTP request returns a Response Object with all the response data (content, encoding, status, etc).

* **pywhatkit**

pywhatkit is a Python library with various helpful features. It's easy-to-use and does not require you to do any additional setup. Currently, it is one of the most popular library for WhatsApp and YouTube automation.

* **wolframalpha**

Wolfram Alpha is an API which can compute expert-level answers using Wolfram's algorithms, knowledgebase and AI technology. It is made possible by the Wolfram Language. This article tells how to create a simple assistant application in Python which can answer simple questions like the ones listed below.

**6.2 Project Modules**

* **main.py**
* **main\_jarvis.py**
* **database.py**
* **speak.py**
* **wakeup.py**
* **wish\_me.py**
* **take\_commans.py**
* **weather.py**
* **whatsapp\_msg.py**
* **wolframalpha\_api.py**

1. **main.py**

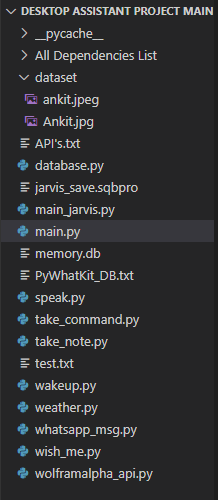
* This is the main module from where the application starts. This module contains Face Recognition System after authentication process it runs main\_jarvis module.

1. **main\_jarvis.py**

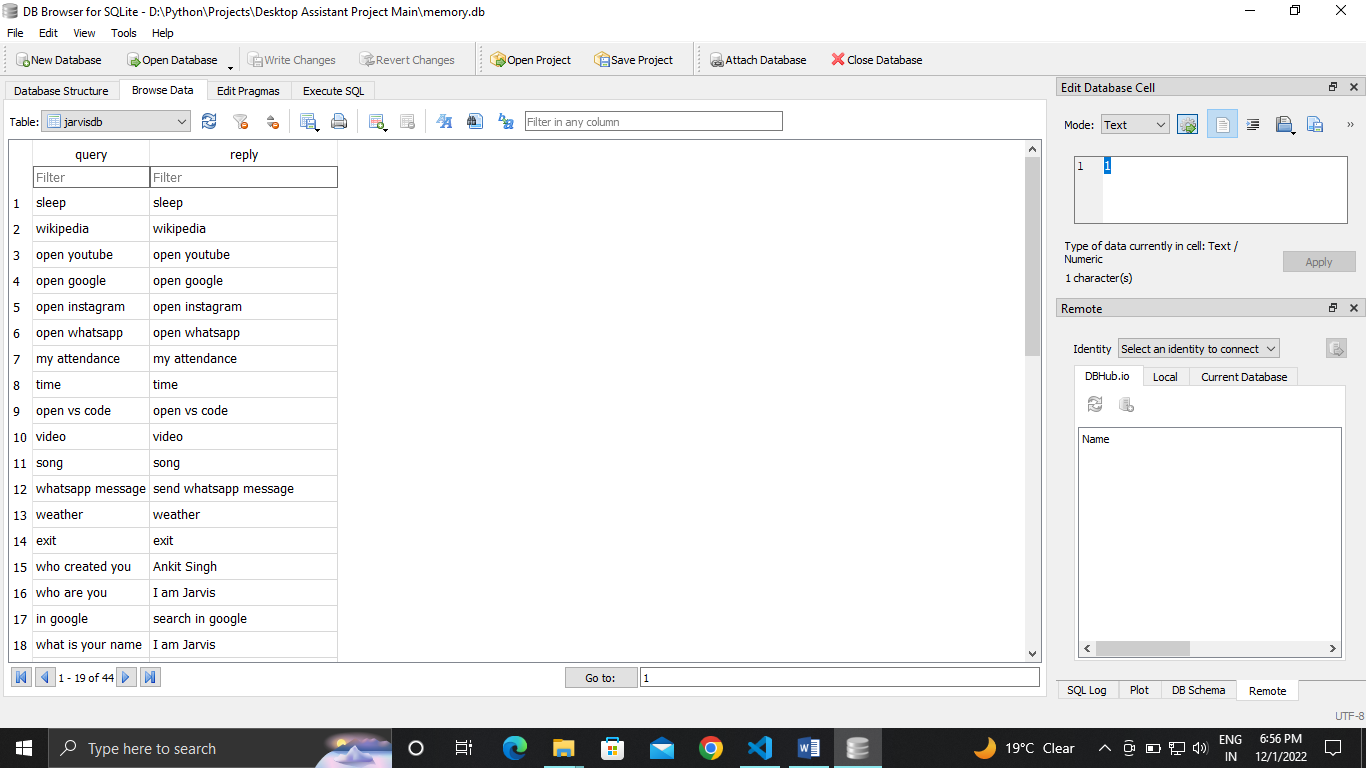
* This is the main module for Voice Assistant it contains main\_execution(), which is the heart of the project. Every query is processed here and accordingly fuctions were called.

1. **Database.py**
   * This module holds all the method related to database connectivity and operations such as insert, delete update query etc.
2. **Speak.py**
   * This module provide capability of speaking to our Virtual Assistant.
3. **wakeup.py**
   * This module has a wakeup() which activate the assistant when user says “hello”.
4. **wish\_me.py**
   * This module has a function which wish the user when he activates the assistant.
5. **take\_command.py**
   * This module is helpful for taking voice commands from user. It has function which recognize voice and convert it into text.
6. **weather.py**
   * This module provide real time weather data for any geographic location and this is done with the API request.
7. **whatsapp\_msg.py**
   * This module helps assistant to automate whatsapp.
8. **wolframalpha\_api.py**
   * This module provide computational intelligence to the assistant all this thing is done with the help of an API**.**

**Screenshot Of All The Modules Of Project**

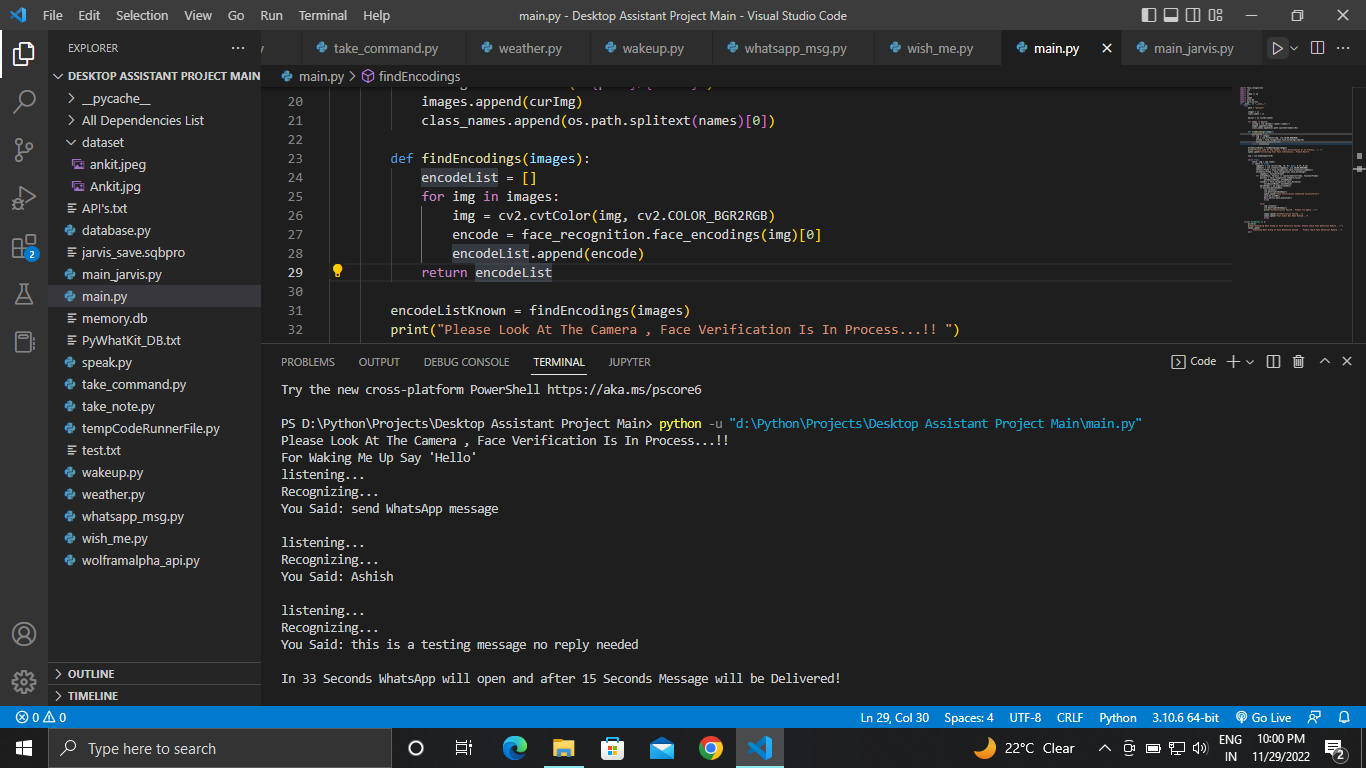
****

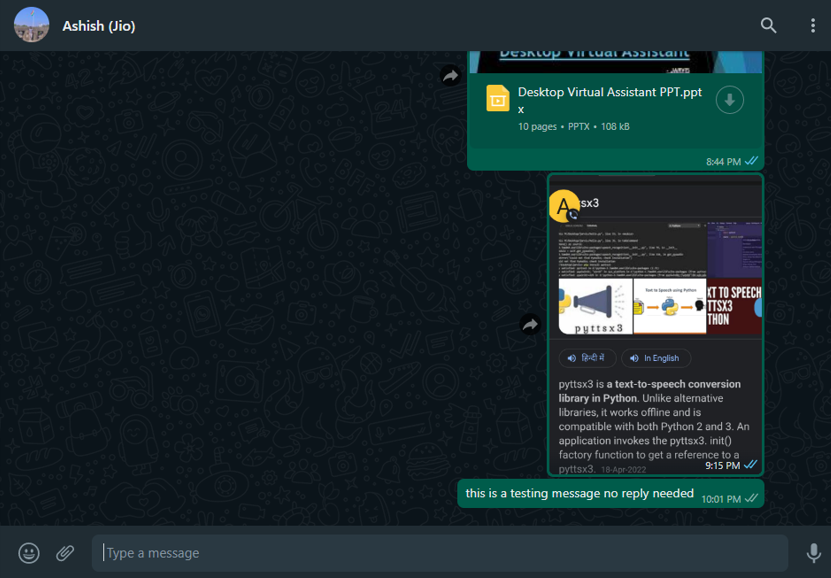
**6.3 Database Design**



* + 1. **Features And Security**
* Opening And Closing Applications
* Web Browsing
* Youtube Searches
* Wikipedia Searches
* Shutdown And Restart PC
* Computational Intelligence
* Live Weather Forecasting
* Whatsapp Automation
* Capability Of Learning New Things In Run Time
* Face Recognition System (Security)

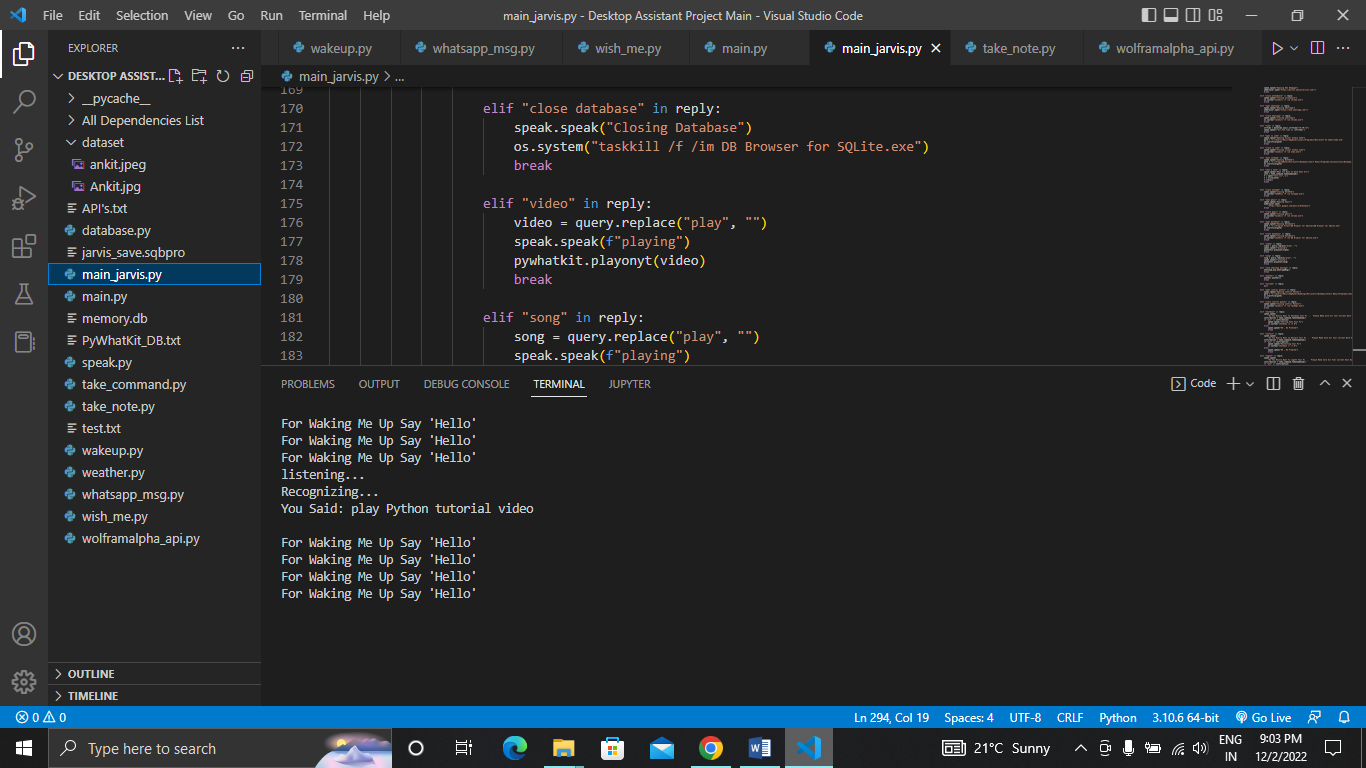
**7.1 Output Screenshots**

**Command**

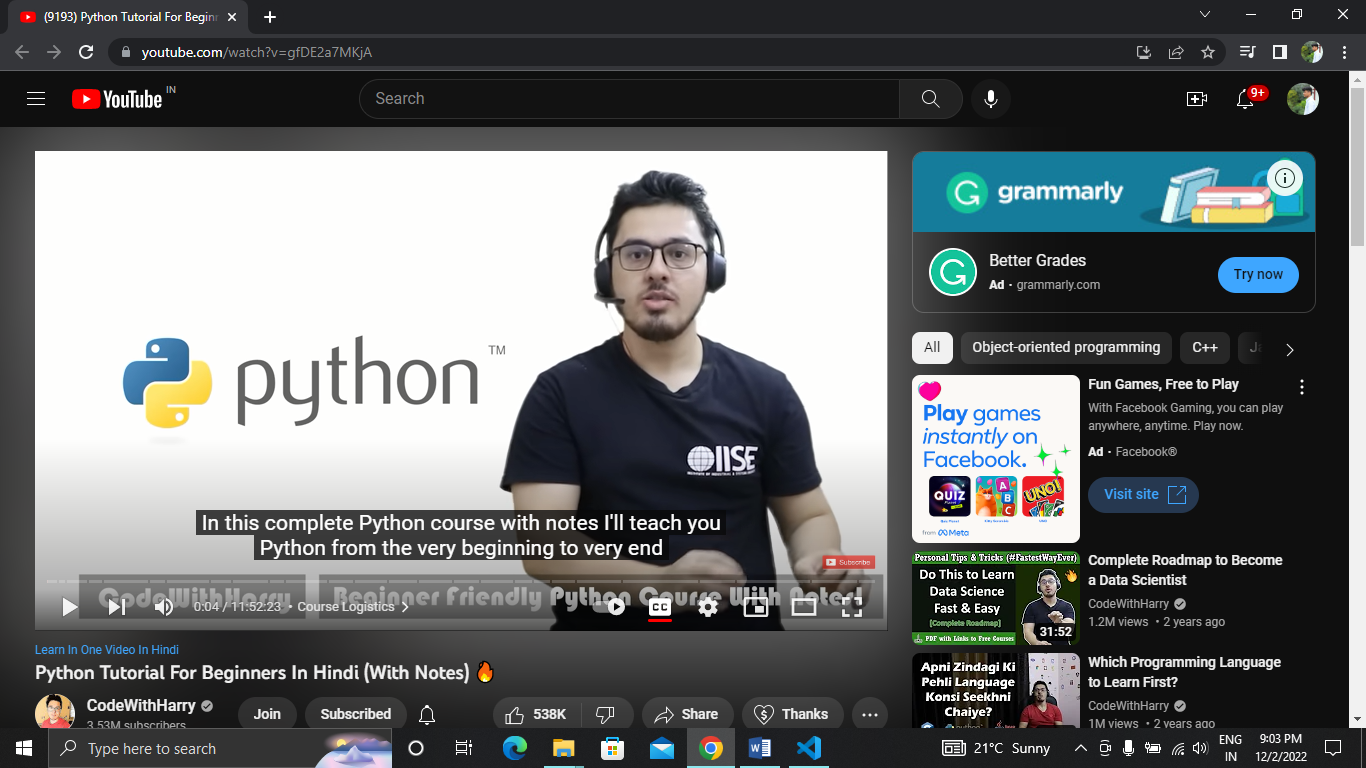
**Output**

**Youtube Searching**

**Command**

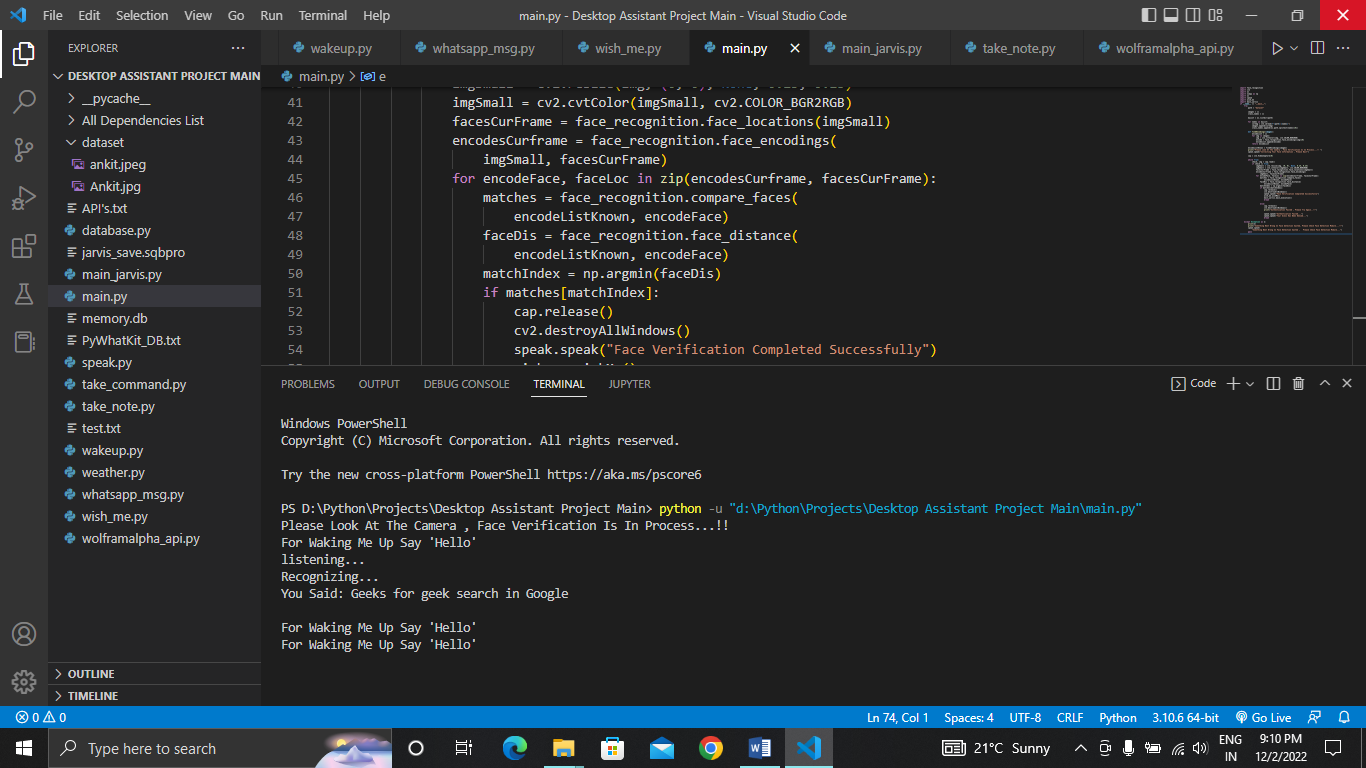
****

**Output**

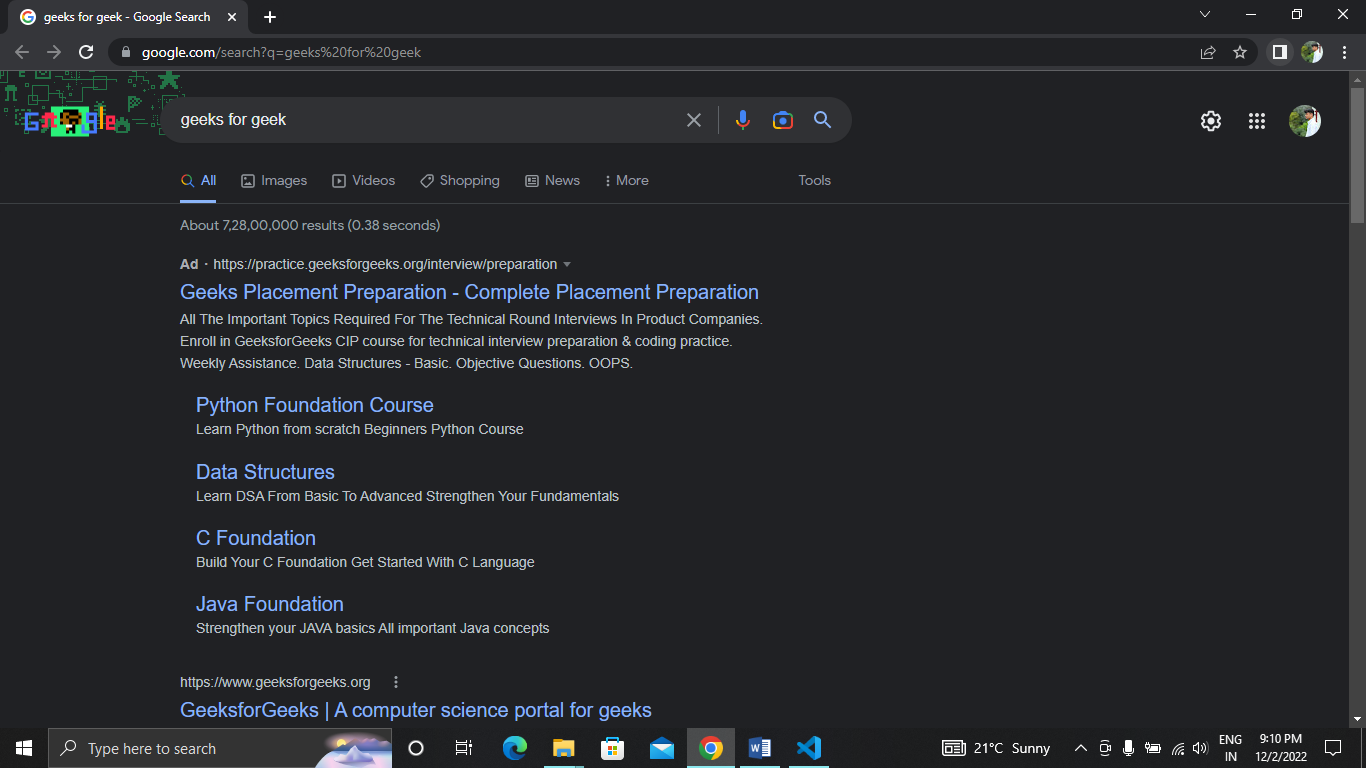
****

**Web Browsing Automation**

**Command**

****

**Output**

****

* + 1. **IMPLEMENTATION OF CODE**
* **main.py**

import face\_recognition

import cv2

import os

import numpy as np

import os

import speak

import wish\_me

import main\_jarvis

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

try:

path = "dataset"

images = []

class\_names = []

myList = os.listdir(path)

for names in myList:

curImg = cv2.imread(f"{path}/{names}")

images.append(curImg)

class\_names.append(os.path.splitext(names)[0])

def findEncodings(images):

encodeList = []

for img in images:

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

encode = face\_recognition.face\_encodings(img)[0]

encodeList.append(encode)

return encodeList

encodeListKnown = findEncodings(images)

print("Please Look At The Camera , Face Verification Is In Process...!! ")

speak.speak("Collecting Your Face Information , Please Wait")

cap = cv2.VideoCapture(0)

while True:

check, img = cap.read()

if check == True:

imgSmall = cv2.resize(img, (0, 0), None, 0.25, 0.25)

imgSmall = cv2.cvtColor(imgSmall, cv2.COLOR\_BGR2RGB)

facesCurFrame = face\_recognition.face\_locations(imgSmall)

encodesCurframe = face\_recognition.face\_encodings(

imgSmall, facesCurFrame)

for encodeFace, faceLoc in zip(encodesCurframe, facesCurFrame):

matches = face\_recognition.compare\_faces(

encodeListKnown, encodeFace)

faceDis = face\_recognition.face\_distance(

encodeListKnown, encodeFace)

matchIndex = np.argmin(faceDis)

if matches[matchIndex]:

cap.release()

cv2.destroyAllWindows()

speak.speak("Face Verification Completed Successfully")

wish\_me.wishMe()

main\_jarvis.main\_execution()

break

else:

cap.release()

cv2.destroyAllWindows()

print("Authentication Failed , Please Try Again..!!")

speak.speak("Authentication Failed...")

speak.speak("Your Acess Has Been Denied...")

break

except Exception as e:

print(e)

print("Something Went Wrong In Face detection System, Please Check Face Detection Module...!!")

speak.speak(

"Something Went Wrong In Face Detection System , Please Check Face Detection Module...")

Pass

* **main\_jarvis.py**

from datetime import datetime

from datetime import timedelta

import wikipedia

import webbrowser

import os

import pywhatkit

import speak

import take\_command

import wakeup

import whatsapp\_msg

import wish\_me

import weather

import database

import os

import wolframalpha\_api

import take\_note

def main\_execution():

try:

con = database.create\_connection()

# wish\_me.wishMe()

while True:

query = wakeup.wakeup().lower()

if "hello" in query:

speak.speak("How Can I Help You..")

while True:

query = take\_command.takeCommanad().lower()

reply = database.reply\_from\_database(query)

if "sleep" in reply:

speak.speak("Ok Sir , You Can Call Me Any Time")

break

elif "wikipedia" in reply:

speak.speak("searching wikipedia...")

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

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

speak.speak("According To Wikipedia ")

print(results)

speak.speak(results)

break

elif "wolframalpha" in reply:

speak.speak("searching...")

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

query = query.strip()

response = wolframalpha\_api.wolframalpha\_search(query)

print(response)

speak.speak(response)

break

elif "open youtube" in reply:

speak.speak("Opening Youtube")

webbrowser.open("https://www.youtube.com/")

break

elif "close youtube" in reply:

speak.speak("Closing Youtube")

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

break

elif "open google" in reply:

speak.speak("Opening Google")

webbrowser.open("https://www.google.co.in/")

break

elif "close chrome" in reply:

speak.speak("Closing google chrome")

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

break

elif "search in google" in reply:

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

search = search.strip()

# print(search)

speak.speak("Searching In Google")

webbrowser.open(

"https://www.google.com/search?q="+search)

break

elif "open instagram" in reply:

speak.speak("Opening Instagram")

webbrowser.open("https://www.instagram.com/")

break

elif "close instagram" in reply:

speak.speak("Closing instagram")

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

break

elif "my attendance" in reply:

speak.speak("Opening Nh2 Webpage")

webbrowser.open("https://portal.aksuniversity.com/")

break

elif "close attendance" in reply:

speak.speak("Closing attendance")

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

break

elif "open whatsapp" in reply:

speak.speak("Opening Whatsapp")

webbrowser.open("https://web.whatsapp.com/")

break

elif "close whatsapp" in reply:

speak.speak("Closing whatsapp")

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

break

elif "time" in reply:

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

speak.speak(f"Sir,The Time Is {strTime}")

break

elif "open vs code" in reply:

speak.speak("Opening Visual Studio Code")

path = "C:\\Users\\DELL\\AppData\\Local\\Programs\\Microsoft VS Code\\Code.exe"

os.startfile(path)

break

elif "close vs code" in reply:

speak.speak("Closing visual studio code")

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

break

elif "open notepad" in reply:

speak.speak("Opening NotePad")

path = "C:\\ProgramData\\Microsoft\\Windows\\Start Menu\\Programs\\Accessories\\Notepad.lnk"

os.startfile(path)

break

elif "take a note" in reply:

speak.speak("What You Want To Note Down Sir")

note = take\_command.takeCommanad()

f = open("test.txt",'w')

f = write(note)

f.close()

break

elif "close notepad" in reply:

speak.speak("Closing NotePad")

os.system("TASKKILL /F /im notepad.exe")

break

elif "open gmail" in reply:

speak.speak("Opening Gmail")

webbrowser.open(

"https://mail.google.com/mail/u/0/#inbox")

break

elif "close gmail" in reply:

speak.speak("Closing gmail")

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

break

elif "open database" in reply:

speak.speak("Opening Database")

path = "C:\\Program Files\\DB Browser for SQLite\\DB Browser for SQLite.exe"

os.startfile(path)

break

elif "close database" in reply:

speak.speak("Closing Database")

os.system("taskkill /f /im DB Browser for SQLite.exe")

break

elif "video" in reply:

video = query.replace("play", "")

speak.speak(f"playing")

pywhatkit.playonyt(video)

break

elif "song" in reply:

song = query.replace("play", "")

speak.speak(f"playing")

pywhatkit.playonyt(song)

break

elif "send whatsapp message" in reply:

whatsapp\_msg.whatsappMsg()

break

elif "weather" in reply:

weather.weather()

break

elif "outside" in reply:

pass

elif "open control pannel" in reply:

speak.speak("Opening Control Pannel")

path = "C:\\Users\\DELL\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\System Tools\\Control Panel.lnk"

os.startfile(path)

break

elif "close control pannel" in reply:

speak.speak("Closing Control Pannel")

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

break

elif "shutdown" in reply:

speak.speak(

"Do You Really Want To Shutdown Your Pc , Please Make Sure All Your Current Work Has Been Saved")

confirmation = take\_command.takeCommanad()

if "yes" in confirmation:

speak.speak("Shuting Down Your Pc")

os.system("shutdown /s /t 0")

else:

speak.speak("OK , No Problem")

break

elif "restart" in reply:

speak.speak(

"Do You Really Want To Restart Your Pc , Please Make Sure All Your Current Work Has Been Saved")

confirmation = take\_command.takeCommanad()

if "yes" in confirmation:

speak.speak("Restarting Your Pc")

os.system("shutdown /r /t 0")

else:

speak.speak("OK , No Problem")

break

elif "logoff" in reply:

speak.speak(

"Do You Really Want To Logoff Your Pc , Please Make Sure All Your Current Work Has Been Saved")

confirmation = take\_command.takeCommanad()

if "yes" in confirmation:

speak.speak("Logging Off Your Pc")

os.system("shutdown /l /t 0")

else:

speak.speak("OK , No Problem")

break

elif "exit" in reply:

speak.speak(

"Thank You For Using Me Sir!! Have A Good Day!!")

exit()

con.close()

else:

if "None" in query:

break

elif len(reply) > 1:

speak.speak(" "+reply)

break

else:

speak.speak(

"I Don't Know This One , Can You Please Tell Me What It Means")

answer = take\_command.takeCommanad()

# reply = database.reply\_from\_database(answer)

if "no" in answer:

speak.speak("OK , No Problem")

break

elif "it means" in answer:

answer = answer.replace("it means", "")

answer = answer.strip()

value = database.reply\_from\_database(answer)

if value == " ":

# database.create\_connection()

# ins="insert into jarvisdb values('"+query+"," +answer+"')"

database.insert\_query\_and\_answer(

query, answer)

speak.speak(

"Thank You For The Answer , I Will Remember It")

break

else:

database.insert\_query\_and\_answer(

query, value)

speak.speak(

"Thank You For The Answer , I Will Remember It")

break

elif "None" in answer:

speak.speak("Please Say That Again Please")

break

except Exception as e:

print(e)

print("Something Went Wrong While Running Main Execution , Please Check Main Execution Module")

speak.speak(

"Something Went Wrong While Running Main Execution , Please Check Main Execution Module")

main\_execution()

* **database.py**

import sqlite3

import speak

# connection=sqlite3.connect("memory.db")

def create\_connection():

connection = sqlite3.connect("memory.db")

return connection

# ins=''' insert into jarvisdb (query,reply) VALUES ('who are you','i am jarvis , a desktop assistent') '''

# connection.execute(ins)

# connection.commit()

def fetch\_query\_and\_reply():

connection = create\_connection()

cursor = connection.cursor()

cursor.execute("SELECT \* FROM jarvisdb")

return cursor.fetchall()

def delete\_who\_what\_query\_and\_reply():

connection = create\_connection()

cursor = connection.cursor()

delete\_who = 'DELETE FROM jarvisdb WHERE query="who"'

delete\_what = 'DELETE FROM jarvisdb WHERE query="what"'

cursor.execute(delete\_who)

cursor.execute(delete\_what)

connection.commit()

connection.close()

def insert\_who\_and\_what\_query\_and\_answer():

connection = create\_connection()

cursor = connection.cursor()

insert\_who = "INSERT INTO jarvisdb(query,reply) VALUES('who','wolframalpha')"

insert\_what = "INSERT INTO jarvisdb(query,reply) VALUES('what','wolframalpha')"

cursor.execute(insert\_who)

cursor.execute(insert\_what)

connection.commit()

connection.close()

def insert\_query\_and\_answer(query, answer):

connection = create\_connection()

cursor = connection.cursor()

insert = "INSERT INTO jarvisdb(query,reply) VALUES('" + \

query+"','"+answer+"')"

# ins= "INSERT INTO jarvisdb (query,reply) VALUES ('who are you','i am jarvis , a desktop assistent')"

cursor.execute(insert)

connection.commit()

connection.close()

def reply\_from\_database(query):

try:

rows = fetch\_query\_and\_reply()

reply = " "

for row in rows:

if row[0].lower() in query.lower():

return row[1]

break

return reply

except Exception as e:

# print(e)

speak.speak("Sorry Something Went Wrong Please Try Again Letter")

insert\_who\_and\_what\_query\_and\_answer()

delete\_who\_what\_query\_and\_reply()

insert\_who\_and\_what\_query\_and\_answer()

* **speak.py**

import pyttsx3

import pyaudio

import speech\_recognition as sr

engine = pyttsx3.init("sapi5")

voices = engine.getProperty("voices")

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

def speak(audio):

engine.say(audio)

engine.runAndWait()

* **wakeup.py**

import speech\_recognition as sr

# import speak

def wakeup():

r = sr.Recognizer()

with sr.Microphone() as source:

r.energy\_threshold = 300

r.pause\_threshold = 2

print("For Waking Me Up Say 'Hello'")

audio = r.listen(source, 0, 5)

try:

query = r.recognize\_google(audio, language="en-in")

# print(f"You Said: {query}\n")

if "hello" not in query:

# print("Say That Again Please")

pass

except Exception as e:

# speak.speak("Sorry Something Went Wrong Please Try Again")

return "None"

return query

* **wish\_me.py**

from datetime import datetime

from datetime import timedelta

import speak

def wishMe():

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

if hour >= 0 and hour < 12:

speak.speak(

"Good Mornig , I'm Jarvis , I Am Here To Assist You , For Waking Me , Say Hello")

elif hour >= 12 and hour < 18:

speak.speak(

"Good Afternoon , I'm Jarvis , I Am Here To Assist You , For Waking Me , Say Hello")

else:

speak.speak(

"Good Evening , I'm Jarvis , I Am Here To Assist You , For Waking Me , Say Hello")

* **take\_commans.py**

import speech\_recognition as sr

def takeCommanad():

r = sr.Recognizer()

with sr.Microphone() as source:

print("listening...")

r.energy\_threshold = 150

r.pause\_threshold = 2

audio = r.listen(source, 0, 4)

try:

print("Recognizing...")

query = r.recognize\_google(audio, language="en-in")

print(f"You Said: {query}\n")

except Exception as e:

print("Say That Again Please...!!")

return "None"

return query

* **weather.py**

import requests

import json

import os

from datetime import datetime

import speak

import take\_command

def weather():

api\_key = os.environ['API\_KEY']

speak.speak("Please Say The City Name To Get Weather Details")

query = take\_command.takeCommanad()

# query = input("city name\n")

if query != "None":

try:

api\_link = "https://api.openweathermap.org/data/2.5/weather?q="+query+"&appid="+api\_key

respons\_obj = requests.get(api\_link)

api\_data = respons\_obj.json()

# print(api\_data)

if api\_data['cod'] == '404':

speak.speak("Please Check City Name And Try Again")

else:

temp\_city = ((api\_data['main']['temp']) - 273.15)

weather\_description = api\_data['weather'][0]['description']

humidity = api\_data['main']['humidity']

wind\_speed = api\_data['wind']['speed']

date\_time = datetime.now().strftime("%H:%M:%S")

print(f"Current Temperature Is :{int(temp\_city)} cel")

print(f"Current Weather Description :{weather\_description}")

print(f"Current Humidity :{humidity} %")

print(f"Current Wind Speed :{wind\_speed} KMPH")

speak.speak("Current Temperature Is" +

str(int(temp\_city))+"degree celcius")

speak.speak("Current Weather Description" +

str(weather\_description))

speak.speak("Current Humidity"+str(humidity)+"Percent")

speak.speak("Current Wind Speed" +

str(wind\_speed)+"Kilo Meter Per Hours")

except Exception as e:

print(e)

Speak.speak(

"Sorry Due To Server Issue Can Not Get Data Please Try Again")

else:

speak.speak("Sorry I Didn't Understand Please Say That Again Please")

* **whatsapp\_msg.py**

from datetime import datetime

from datetime import timedelta

import pywhatkit

import speak

import take\_command

contact = {"ashish": "+916263343153",

"abhijeet": "+919425146509", "papa": "+919752952515"}

def whatsappMsg():

try:

speak.speak("Please Tell Me The Recipient Name")

to = take\_command.takeCommanad().lower()

speak.speak("What Message You Want To Sent")

content = take\_command.takeCommanad()

for key in contact:

if key == to:

number = contact[key]

break

time\_hour = int(datetime.now().strftime("%H"))

time\_min = int(

(datetime.now()+timedelta(minutes=1, seconds=15)).strftime("%M"))

pywhatkit.sendwhatmsg(number, content, time\_hour, time\_min)

except Exception as e:

print("Sorry This Message Would Not Be Send , Please Try Again ")

* **wolframalpha\_api.py**

import wolframalpha

import os

import speak

import take\_command

import database

api\_id = os.environ['Wolframalpha\_API']

def wolframalpha\_search(query):

try:

client = wolframalpha.Client(api\_id)

response = client.query(query)

output = str(next(response.results).text)

return output

except Exception as e:

speak.speak(

"I Don't Know This One , Can You Please Tell Me What It Means")

answer = take\_command.takeCommanad()

if "no" in answer:

speak.speak("OK , No Problem")

elif "it means" in answer:

answer = answer.replace("it means", "")

answer = answer.strip()

value = database.reply\_from\_database(answer)

if value == " ":

database.delete\_who\_what\_query\_and\_reply()

database.insert\_query\_and\_answer(query, answer)

speak.speak("Thank You For The Answer , I Will Remember It")

database.insert\_who\_and\_what\_query\_and\_answer()

else:

database.delete\_who\_what\_query\_and\_reply()

database.insert\_query\_and\_answer(query, value)

speak.speak("Thank You For The Answer , I Will Remember It")

database.insert\_who\_and\_what\_query\_and\_answer()

**9.REFERENCES**

**Website References**

1. <https://docs.python.org>
2. <https://www.python.org>
3. <https://www.geeksforgeeks.org>
4. <https://openweathermap.org>
5. <https://www.wolframalpha.com>
6. <https://www.wikipedia.org>

**Youtube Channel References**

1. CodeWithHarry
2. Avi Upadhayay

**10.CONCLUSION**

Through this voice assistant, we have automated various services using voice commands. It eases most of the tasks of the user like searching the web, retrieving weather forecast details, opening and closing applications, sending messages on whatsapp etc. Our aim is to make a voice assistant which can do most of the desktop work with voice commands. The future plan is to integrate **Artificial Intelligence** in our voice assistant project. The destiny of these assistants may have the virtual assistants integrated with Artificial Intelligence which incorporates Machine Learning, Neural Networks and IoT. With this technology, we are able to gain new heights.

In future our plan is to integrate Artificial Intelligence And Machine Learning Algorithms to the assistant after this integration it will capable of taking decisions and it can understand our emotions also.

This is the first step in the new era of automation in future almost every task can be done through the voice command apart from desktop and devices assistant can also automates home appliance such as TV, AC, Lights, Fan etc.

Hence this is technology that can plays an important role in the Automation Era.