PROJECT REPORT

ON

"A.I. ASSISTANT"

SUBMITTED BY

Mr. Harsh Nandlal Yadav

Mr. Vijay Narkar Sambhaji

Mr. Sahil Shekhar Kadam

Mr. Jitesh Ramdas Gawde

GUIDED BY

PROF. M.Y.CHAUDHARI

DEPT. OF COMPUTER ENGINEERING

DEVI MAHALAXMI POLYTECHNIC COLLEGE, TITWALA

2022-23

CERTIFICATE

This is to certify that the Project Entitled

"A.I. ASSISTANT"

Submitted by

Harsh Nandlal Yadav
Sahil Shekhar Kadam
Vijay Narkar Sambhaji
Jitesh Ramdas Gawde

In the partial fulfillment of Diploma in ***Computer Engineering***

Recognized by Maharashtra State Board Of Technical Education,
for the Academic Year 2022 - 23.

GUIDED BY

PROF. M.Y.CHAUDHARI

HEAD OF DEPT.

PROF. M.Y.CHAUDHARI

EXTERNAL EXAMINER

PRINCIPAL

DR. M.B.ASRE

DEVI MAHALAXMI POLYTECHNIC COLLEGE, TITWALA

COMPUTER ENGINEERING

2022-23

APPROVED CERTIFICATE

This is to certify that

Harsh Nandlal Yadav
Sahil Shekhar Kadam
Vijay Narkar Sambhaji
Jitesh Ramdas Gawde

Of final year have satisfactorily completed the project entitled

“A.I. ASSISTANT”

And accomplished the presentation on the same for the session 2022-2023.

I am satisfied, with the work of this project team.

With lot of efforts my project team has completed their project with title

*“***A.I. ASSISTANT***”.*

They regularly contacted me for suggestion and advice through out the project work as well as for project documentation also.

DMPOLY

APPROVED BY
PROF. M.Y.CHAUDHARI

DEVI MAHALAXMI POLYTECHNIC COLLEGE, TITWALA

COMPUTER ENGINEERING

2022-23

ACKNOWLEGEMENT

We feel great pleasure in expressing our deepest sense of gratitude and sincere thanks to our guide **Prof. Mr. M.Y.Chaudhari** for her valuable guidance during the Project work, without which it would have been very difficult task. We have no words to express my sincere thanks for valuable guidance, extreme assistance and cooperation extended to all the **Staff Members** of our Department.

This acknowledgement would be incomplete without expressing our special thanks to **“Prof. Mr. M.Y.Chaudhari”** HOD of **“Computer Engineering”** for his support during the work.

Last but not least. We would like to thanks all the Teaching. Non Teaching staff members of the my/our Department and my colleagues those who helped us directly or indirectly for completing of this Project successfully.

Project Team :
Mr. Harsh Nandlal Yadav
Mr. Sahil Shekhar Kadam
Mr. Vijay Narkar Sambhaji
Mr. Jitesh Ramdas Gawde

CONTENT

1.INTRODUCTION

INTRODUCTION

In today's era almost all tasks are digitalized. We have Smartphone in hands and it is nothing less than having world at your fingertips. These days we aren't even using fingers. We just speak of the task and it is done. There exist systems where we can say Text Dad, "I'll be late today." And the text is sent. That is the task of a Voice Assistant. It also supports specialized task such as booking a flight, or finding cheapest book online from various ecommerce sites and then providing an interface to book an order are helping automate search, discovery and online order operations.

Voice Assistants are software programs that help you ease your day-to-day tasks, such as showing weather report, creating reminders, making shopping lists etc. They can take commands via text (online chat bots) or by voice. Voice based intelligent assistants need an invoking word or wake word to activate the listener, followed by the command. We have so many virtual assistants, such as Apple's Siri, Amazon's Alexa and Microsoft's Cortana.

This system is designed to be used efficiently on desktops. Personal assistant software improves user productivity by managing routine tasks of the user and by providing in-formation from online sources to the user. It is effortless to use.

Voice searches have dominated over text search. Web searches conducted via mobile devices have only just overtaken those carried out using a computer and the analysts are already predicting that 50% of searches will be via voice by 2020.

Virtual assistants are turning out to be smarter than ever. Allow your intelligent assistant to make email work for you. Detect intent, pick out important information, automate processes, and deliver personalized responses.

This project was started on the premise that there is sufficient amount of openly available data and information on the web that can be utilized to build a virtual assistant that has access to making intelligent decisions for routine user activities.

1.1 Background

Voice assistants are software applications that use natural language processing (NLP), machine learning, and artificial intelligence (AI) to recognize and respond to voice commands or queries. They are designed to simulate human-like interactions, making it easier for users to interact with devices or software applications using voice commands. The first voice assistant was introduced in 2011 by Apple as Siri, which became popular and triggered the growth of voice assistants in various devices and applications.

Over time, voice assistants have become a common feature in various devices such as smartphones, smart speakers, home automation devices, and in-car infotainment systems. They have also been integrated into applications such as messaging apps, e-commerce platforms, and productivity tools. The voice assistant market is dominated by major technology players such as Amazon, Google, Microsoft, and Apple, but other companies are also investing in the technology.

Voice assistants are continually evolving and improving, with advancements in natural language processing, deep learning, and speech recognition technology. They have become increasingly accurate and can now handle complex requests and perform a wide range of tasks such as playing music, setting reminders, answering questions, and even controlling other smart devices in the home. As a result, they are changing the way we interact with technology, making it more convenient and user-friendly.

1.2 Purpose:

Purpose of virtual assistant is to being capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news. Virtual assistant sense able users to speak natural language voice commands in order to operate the device and its apps. There is an increased overall awareness and a higher level of comfort demonstrated specifically by millennial consumers. In this ever-evolving digital world where speed, efficiency, and convenience are constantly being optimized, it's clear that we are moving towards less screen interaction.

1.3 Objectives

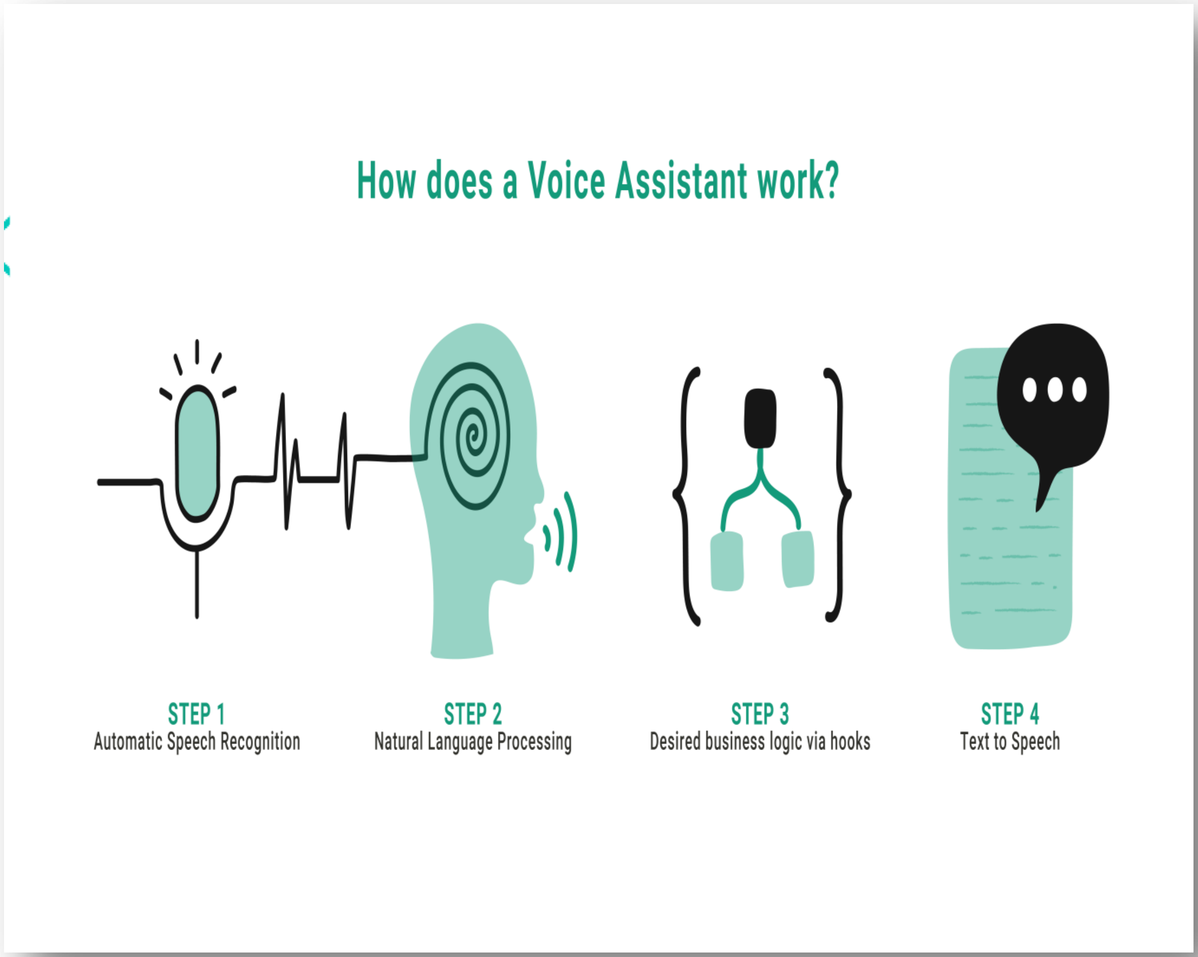
Main objective of building personal assistant software (a virtual assistant) is using semantic data sources available on the web, user generated content and providing knowledge from knowledge databases. The main purpose of an intelligent virtual assistant is to answer questions that users may have. This may be done in a business environment, for example, on the business website, with a chat interface. Virtual assistants can tremendously save you time. We spend hours in online research and then making the report in our terms
of understanding.

One of the main advantages of voice searches is their rapidity. In fact, voice is reputed to be four times faster than a written search: whereas we can write about 40 words per minute, we are capable of speaking around 150 during the same period of time15. In this respect, the ability of personal assistants to accurately recognize spoken words are a prerequisite for them to be adopted by consumers.

2.Overview of Voice Assistants

2.1 Definition of Voice Assistants

Voice assistants are artificial intelligence-powered software applications that use natural language processing (NLP) to understand and respond to voice commands and queries. These applications are designed to interact with humans in a conversational manner, providing information, completing tasks, and even controlling devices and appliances through voice commands.


How does a Voice Assistant work?
STEP 1
Automatic Speech Recognition
STEP 2
Natural Language Processing
STEP 3
Desired business logic via hooks
STEP 4
Text to Speech

Voice assistants typically use machine learning algorithms and deep neural networks to recognize and interpret spoken language, and they can perform a wide variety of tasks, from answering simple questions to ordering products, setting reminders, playing music, and even making phone calls.

Some of the most popular voice assistants include Amazon's Alexa, Google Assistant, Apple's Siri, and Microsoft's Cortana. These assistants are often integrated into smart speakers, smartphones, and other devices, allowing users to interact with them hands-free and with ease.

2.2 Types of Voice Assistants

There are several types of voice assistants, each with its own set of features and capabilities. Here are some of the most common types:

1. **Smart speakers :** These are standalone devices that are primarily designed to interact with voice assistants. They often come with high-quality speakers and can be used to play music, set alarms, control smart home devices, and more.

2. **Mobile assistants:** These are voice assistants that are integrated into mobile devices, such as smartphones and tablets. They can be used to perform a wide range of tasks, including making phone calls, sending messages, scheduling appointments, and more.

3. **Personal assistants:** These are voice assistants that are designed to be more personalized to the user's needs and preferences. They can be customized to perform specific tasks, such as ordering food, booking flights, or playing specific types of music.

4. **In-car assistants:** These are voice assistants that are designed to be used while driving. They can be used to make phone calls, send text messages, and get directions, all without taking your hands off the steering wheel.

5. **Voice-activated appliances:** These are voice assistants that are integrated into household appliances, such as refrigerators, washing machines, and ovens. They can be used to control the appliance, set timers, and even order groceries.

These are just a few examples of the different types of voice assistants available today. As the technology continues to evolve, we can expect to see even more advanced and sophisticated voice assistants in the future.

2.3 Brief History of Voice Assistants:

The history of voice assistants can be traced back to the early days of computing, when researchers began exploring ways to enable computers to understand and respond to human speech. Here's a brief overview of some key milestones in the history of voice assistants:

1. **The first speech recognition system:** The first speech recognition system was developed in the 1950s by Bell Labs, but it was limited to recognizing only a few words.

2. **Early voice assistants:** In the 1980s and 1990s, a number of early voice assistants were developed, including Apple's Newton, IBM's Simon, and Dragon Systems' Dragon NaturallySpeaking.

3. **Siri:** In 2011, Apple introduced Siri, a voice assistant for its iPhone that was capable of answering questions, making recommendations, and performing tasks.

4. **Google Assistant:** In 2016, Google introduced its voice assistant, Google Assistant, which was integrated into its Pixel phones and Google Home smart speakers.

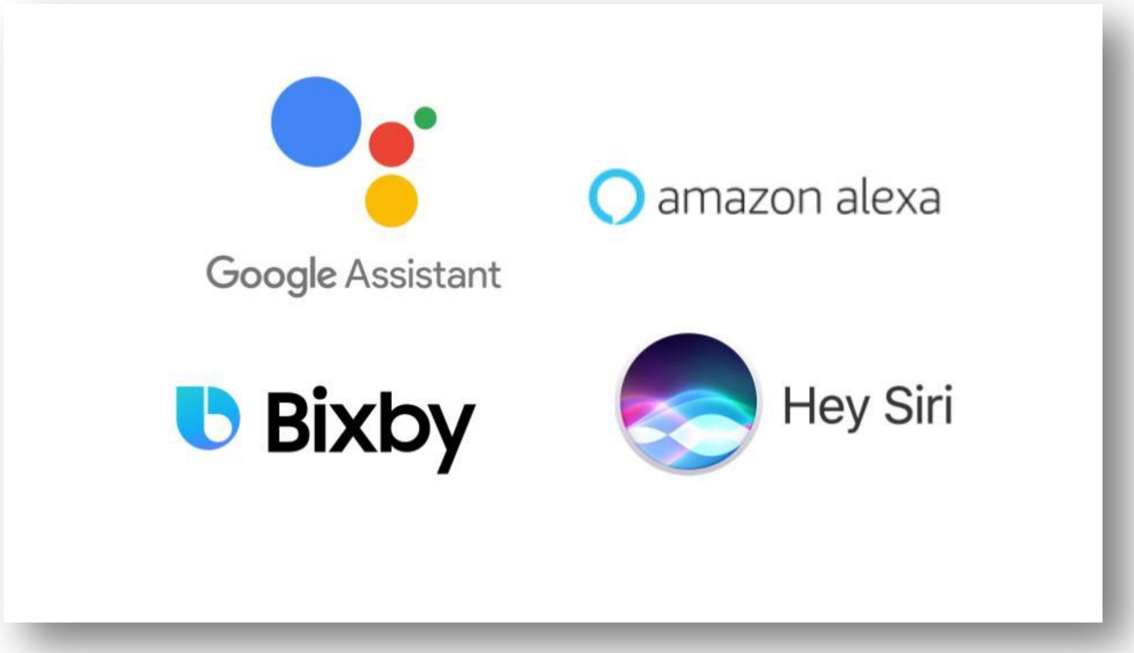
5. **Amazon Alexa:** Also in 2016, Amazon introduced Alexa, a voice assistant that was integrated into its Echo smart speaker.

6. **Expansion to other devices:** Since then, voice assistants have been integrated into a wide range of devices, including smart TVs, cars, and home appliances.

Today, voice assistants are a ubiquitous part of our daily lives, and they continue to evolve and improve as developers and researchers explore new ways to make them more intuitive and helpful.

2.4 Major Voice Assistant Providers

There are several major providers of voice assistants in the market today, each with its own strengths and features. Here are some of the most popular ones:



1. **Amazon Alexa:** Amazon's Alexa is a popular voice assistant that is integrated into its Echo smart speakers and a wide range of other devices. Alexa is known for its wide range of skills and the ability to control smart home devices.

2. **Google Assistant:** Google's Assistant is another popular voice assistant that is integrated into a range of devices, including smartphones, smart speakers, and smart displays. It is known for its ability to provide personalized recommendations and assistance based on user preferences.

3. **Apple Siri:** Apple's Siri is a voice assistant that is integrated into its iOS devices, including iPhones, iPads, and Mac computers. Siri is known for its seamless integration with other Apple products and its ability to perform tasks such as setting reminders, sending messages, and making phone calls.

4. **Microsoft Cortana:** Microsoft's Cortana is a voice assistant that is integrated into its Windows operating system, as well as its Surface devices and other products. Cortana is known for its productivity features, including the ability to schedule appointments and set reminders.

5. **Samsung Bixby**: Samsung's Bixby is a voice assistant that is integrated into its Galaxy smartphones and other devices. Bixby is known for its ability to perform device-specific tasks, such as taking screenshots and adjusting settings.

These are just a few examples of the major voice assistant providers in the market today. As the technology continues to evolve, we can expect to see even more players entering the space and further innovation in voice assistant capabilities.

3.Design and Development

3.1 Algorithm:

Algorithm for the developing A.I. Assistant:

1. Import all the required modules:
    - tkinter
    - filedialog
    - messagebox
    - schedule
    - smtplib
    - time
    - requests
    - datetime
    - openai
    - speech_recognition
    - pyttsx3
    - webbrowser
    - pywhatkit
    - pyautogui
    - cv2
2. Initialize the openai API key and text-to-speech engine.
3. Define a function named "send_email()" that takes email_list, subject_entry, and body_text as parameters. Inside the function, set the sender_email and sender_password. Use the try and except blocks to catch any exception and show the error message in a message box. If the email is sent successfully, show a success message in a message box.
4. Define a function named "open_popup()" that creates a new window using the "Toplevel()" method. Inside the function, create entry boxes, labels, and buttons for recipient emails, subject, message body, time, and send/schedule email. In the "schedule_email()" function, get the scheduled_time from the time_entry and de-

fine a nested function “send_scheduled_email()” that calls the “send_email()” function to send the email to the recipient. Use the “schedule.every().day.at()” method to schedule the email at the specified time. Use the while loop to check for any pending scheduled emails using the “schedule.run_pending()” method and sleep for 1 second.

5. Define a function named “Photo()” that captures the video using the “cv2.VideoCapture()” method and displays the captured video in a new window using the “cv2.imshow()” method. Use the “cv2.waitKey()” method to wait for the user to press a key. If the user presses the escape key, the application will be closed. If the user presses the space bar, take a screenshot using the “cv2.imwrite()” method and save it with a unique name.
6. Define a function named “take_screenshot()” that uses the pyautogui.screenshot() method to take a screenshot and opens a file dialog to choose a directory to save the screenshot. If a directory is chosen, save the screenshot to that directory with a unique name.
7. Define a function named “ai_response()” that takes a user_question as input, uses the OpenAI API to get the response, and returns the generated response.
8. Define a function named “speak()” that takes text as input, uses the pyttsx3 module to convert the text to speech, and speaks the text.
9. Define a function named “reply()” that gets the “user_question” from the “qField” entry box, appends it to the “textArea” text widget, and checks for various commands like play a song on YouTube, open YouTube, open Google, open Yahoo, etc. If the user asks to play a song, extract the song name from the user_question and use the “pywhatkit.playonyt()” method to play the song on YouTube. If the user asks to open YouTube, use the “webbrowser.open()” method to open YouTube. If the user asks to take a screenshot, call the “take_screenshot()” function. If the user asks a general question, call the “ai_response()” function to get the AI-generated response and speak it using the “speak()” function.
10. Define a function called “speechRecognition” that takes no arguments.
11. Create a “SpeechRecognizer” object called r.

12. Use a try-except block to catch any errors that may occur during speech recognition.
13. Use the with statement to open the default system microphone as a source of audio input.
14. Print "Listening..." to the console.
15. Set the pause threshold for the “SpeechRecognizer” to 1 second to ensure that the recognizer stops listening when the user is finished speaking.
16. Call the “adjust_for_ambient_noise” method of the SpeechRecognizer object to adjust for background noise during audio input.
17. Use the listen method of the SpeechRecognizer object to capture audio input from the microphone.
18. Print "Recognizing..." to the console.
19. Call the “recognize_google” method of the SpeechRecognizer object to recognize the audio input and convert it to text using the Google Speech Recognition API.
20. Pass the audio variable and specify the language as "en-in".
21. Print the recognized text to the console.
22. Clear the contents of the “qField” Entry widget.
23. Insert the recognized text into the “qField” Entry widget.
24. Call the reply function to generate a response based on the user's query.
25. If an error occurs during speech recognition, print the error message to the console and return the string "None".
26. Return the recognized text.
27. Define a function named get_weather_data that takes a city as an argument.
28. Inside the function, set the api_key variable to your own OpenWeatherMap API key.
29. Set the url variable to the OpenWeatherMap API endpoint with the city argument and the api_key.
30. Use the requests module to make a GET request to the url.

31. If the response status code is 200 (i.e., the request was successful), parse the response JSON data and extract the temperature, feels like temperature, and weather description data.
32. Return a formatted string that contains the current weather data for the given city.
33. If the response status code is not 200, return an error message with the response status code.
34. Define a function named show_weather that takes a city as an argument.
35. Inside the function, call the get_weather_data function with the city argument to get the weather data for the given city.
36. Insert the weather data into a textArea widget and speak the weather data using the speak function.
37. Define a function named show_weather_popup that shows a popup window with an input field for the user to enter a city.
38. Inside the function, create a Toplevel widget and set its geometry and title.
39. Create a Label widget and an Entry widget for the user to enter a city.
40. Create a Button widget that calls the show_weather function with the city entered by the user.
41. Pack the Label, Entry, and Button widgets into the popup window

3.2 Features

- **Email scheduling:** The personal assistant allows users to schedule emails to be sent automatically at a specified time using the schedule and smtplib modules. The email content can be entered in a pop-up window along with the recipient email addresses and subject.
- **Voice recognition:** The speech_recognition module is used to recognize the user's voice input and perform the desired task. The pyttsx3 module is used to convert the assistant's response to speech output.

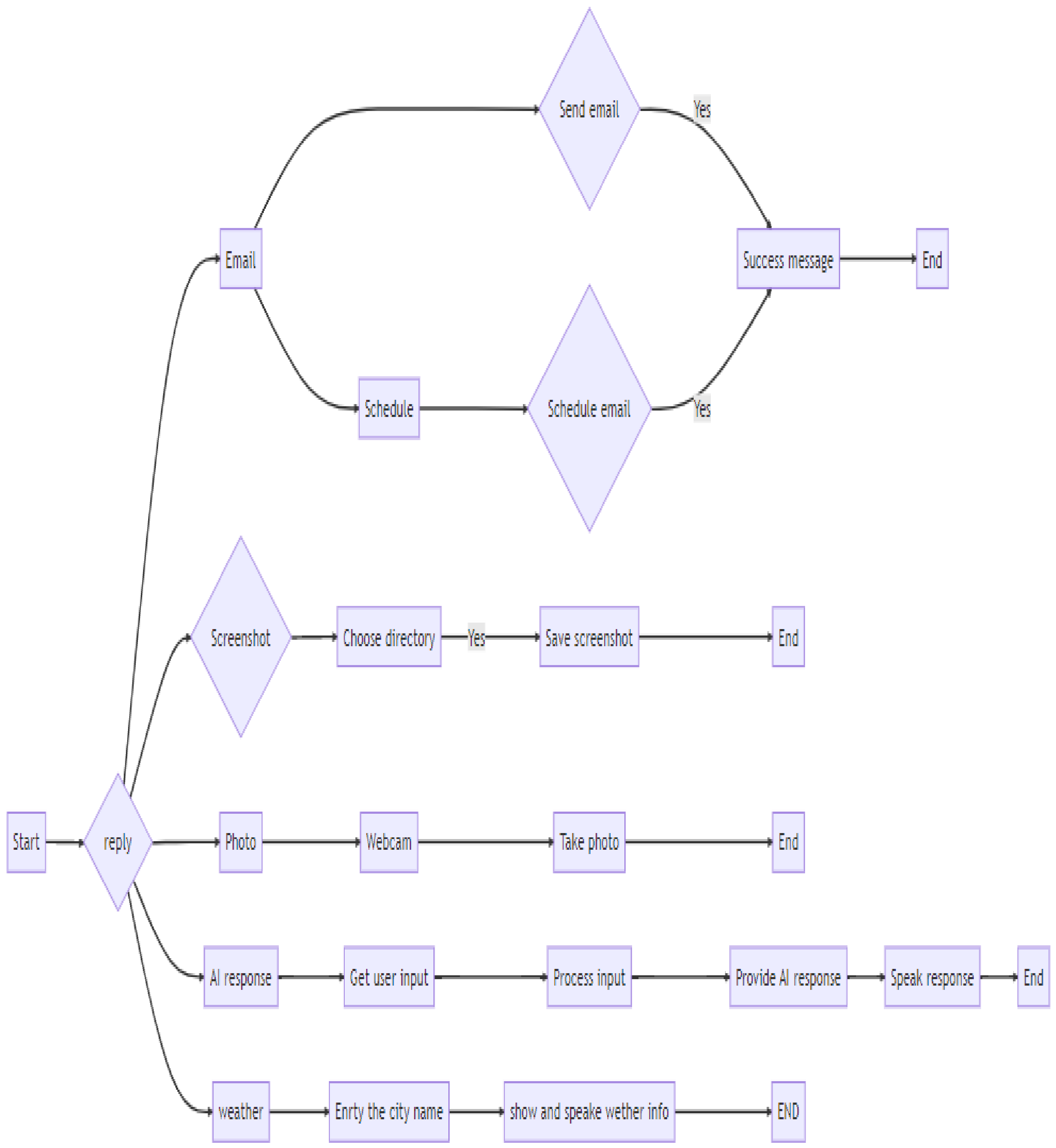
- **Web browsing:** The assistant can open a web browser and visit the desired website using the webbrowser module. The assistant can also open YouTube, Google, or Yahoo on the user's command.
- **AI response:** The OpenAI module is used to generate a response to user queries using a pre-trained language model. The model can provide a response to a wide range of queries, including factual information, mathematical calculations, and even jokes.
- **Image capturing:** The personal assistant can capture an image using the computer's webcam and save it to a file using the OpenCV module. The assistant can also take screenshots of the user's screen using the pyautogui module.
- **Create a simple program:** The OpenAI module we can create simple program on language like HTML, Python, CSS, C, C++ ,JAVA, etc.
- **Create a simple PPT:** By using OpenAI module we create simple power point presentation on any topic in a few second.

3.3 Tools and Technologies used:

- **Python:** Python is a high-level programming language that is used to develop various applications. It is a popular choice among developers due to its simplicity and versatility.
- **Tkinter:** Tkinter is a Python library that is used to develop graphical user interfaces (GUI). It is a standard GUI library for Python and is included with most Python installations.
- **schedule:** Schedule is a Python library that allows scheduling of tasks to be executed at specific times or intervals.
- **smtplib:** Smtplib is a Python library that is used to send emails using the Simple Mail Transfer Protocol (SMTP).
- **requests:** Requests is a Python library that is used to make HTTP requests.
- **datetime:** The datetime module supplies classes for working with dates and times.
- **OpenAI:** OpenAI is an artificial intelligence research laboratory consisting of the for-profit corporation OpenAI LP and its parent company, the non-profit OpenAI Inc. The code uses OpenAI's text-based AI model to generate responses to user inputs.

- **SpeechRecognition:** SpeechRecognition is a Python library that is used to recognize speech.
- **Pyttsx3:** Pyttsx3 is a Python library that is used to convert text to speech.
- **webbrowser:** The webbrowser module is used to open web pages in a browser.
- **pywhatkit:** pywhatkit is a Python library that is used to perform several tasks like sending WhatsApp messages, searching on Google, and playing YouTube videos.
- **pyautogui:** pyautogui is a Python library that is used to perform various automation tasks like taking screenshots and controlling the mouse and keyboard.
- **cv2:** cv2 is a Python library that is used for computer vision applications, specifically for image and video processing.
- **Processor:** Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30 GHz
- **RAM:** 4GB RAM
- **Operating System:** Windows 10 Pro

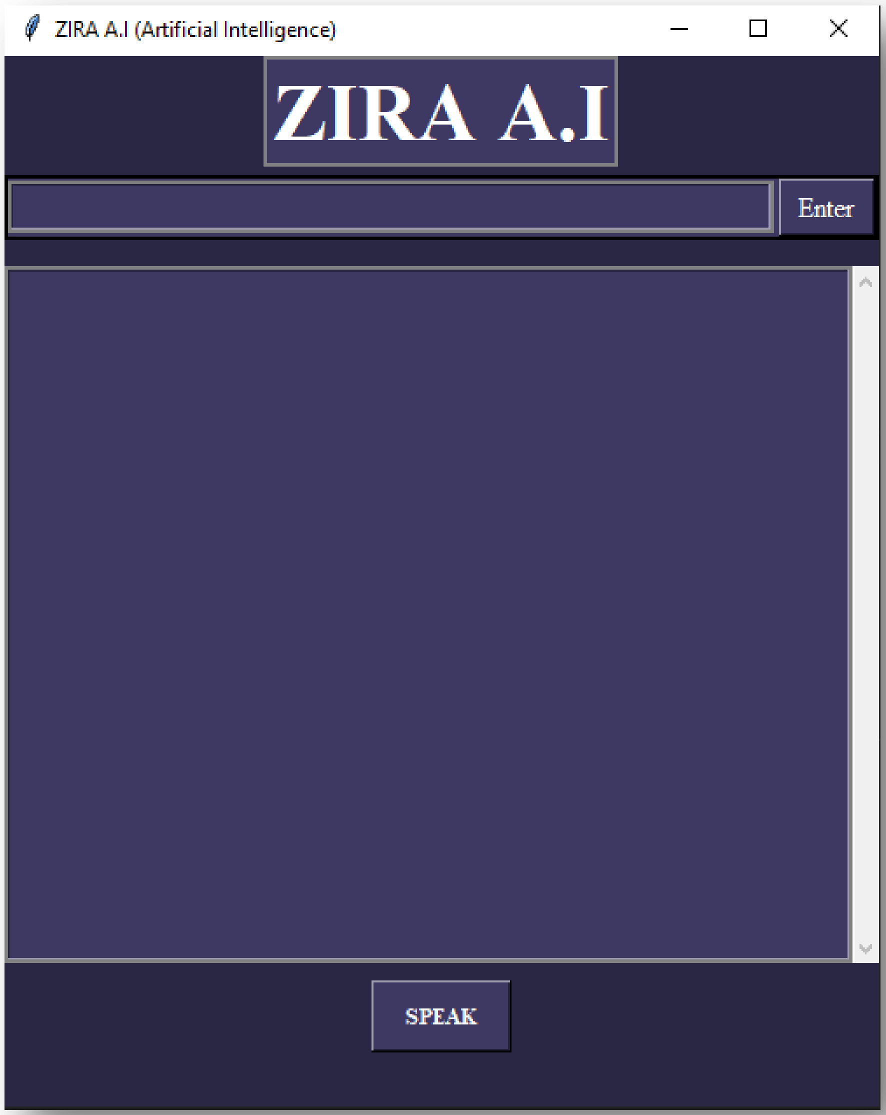
3.4 Basic Flowchart



3.4.1 Diagram of basic flowchart

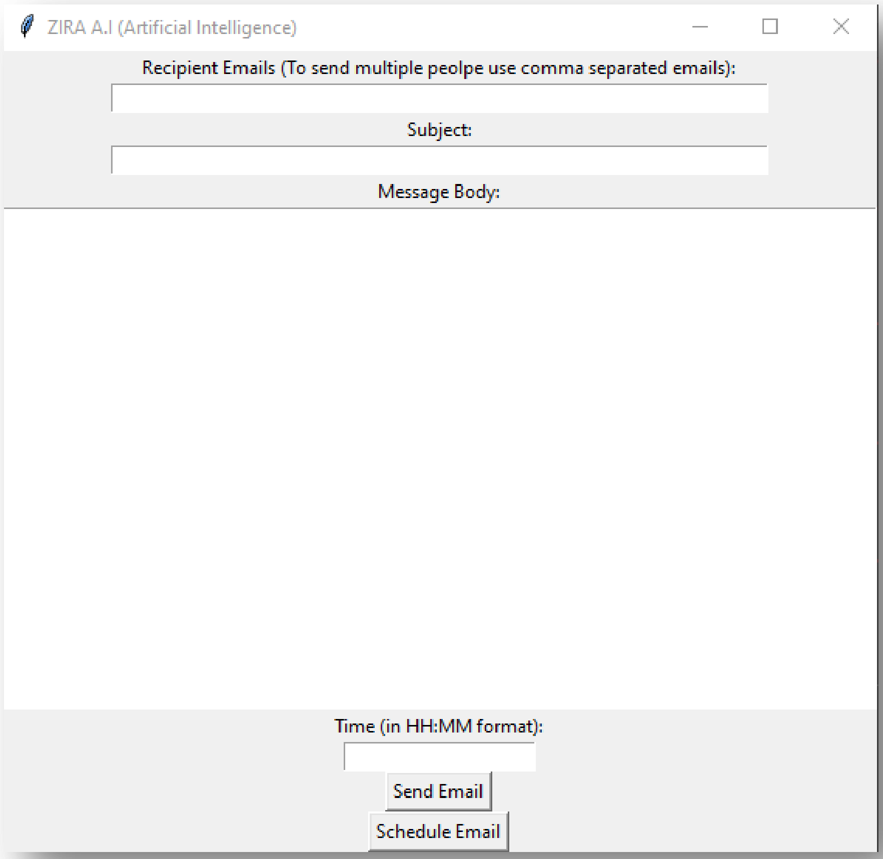
3.5 Interface Design in the project:

- The project utilizes the Tkinter library to design a graphical user interface for the various functions. The user interface is designed to be user-friendly and intuitive, with clear labels and instructions for each function.



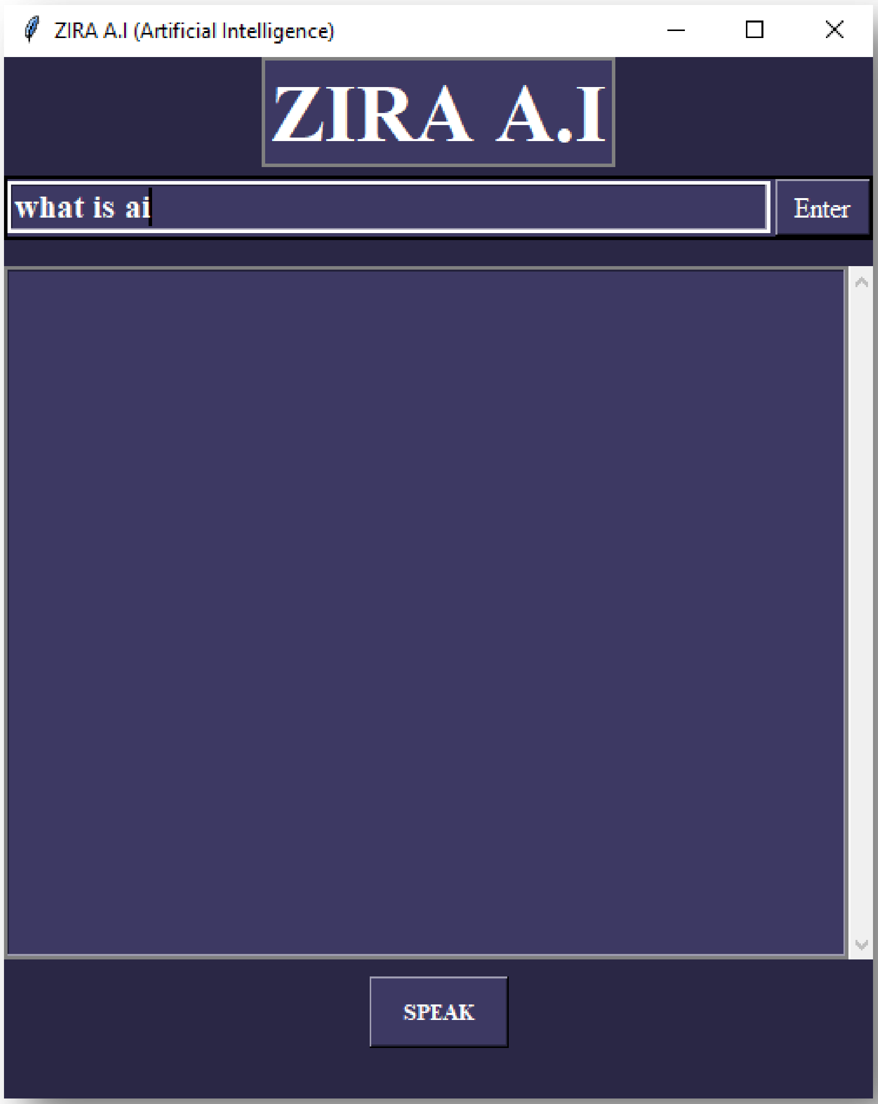
3.5.1 GUI window

- When the user runs the program, a GUI window is opened which provides the user with several options. The user can choose to take a screenshot of their screen, send an email, schedule an email to be sent at a later time, open a web page, use speech recognition to ask a question and get an AI response, or take a photo using their webcam.
- The "Send Email" option opens a pop-up window that allows the user to enter the recipient's email address, the email subject, and the email body. Once the user has entered this information, they can click on the "Send Email" button to send the email.
- The "Schedule Email" option allows the user to schedule an email to be sent at a specific time. The user enters the recipient's email address, the email subject, and the email body, along with the time they want the email to be sent. The program then uses the schedule library to schedule the email to be sent at the specified time.

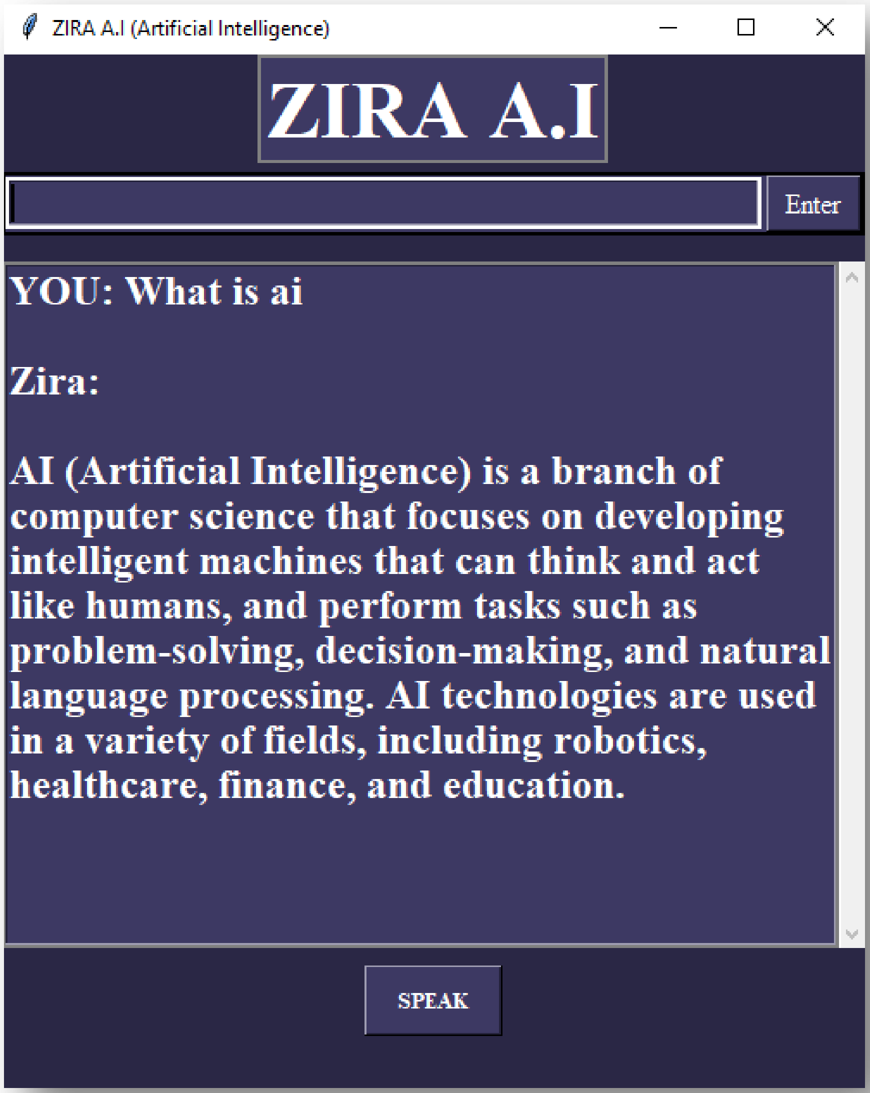

ZIRA A.I (Artificial Intelligence)
Recipient Emails (To send multiple peolpe use comma separated emails):
Subject:
Message Body:
Time (in HH:MM format):
Send Email
Schedule Email

3.5.2 Email GUI

- The "Ask Question" option allows the user to speak into their microphone and ask a question. The program uses the speech recognition library to convert the user's speech into text, which is then sent to the OpenAI API to get a response. The response is then displayed in the text box.



3.5.3 Enter question



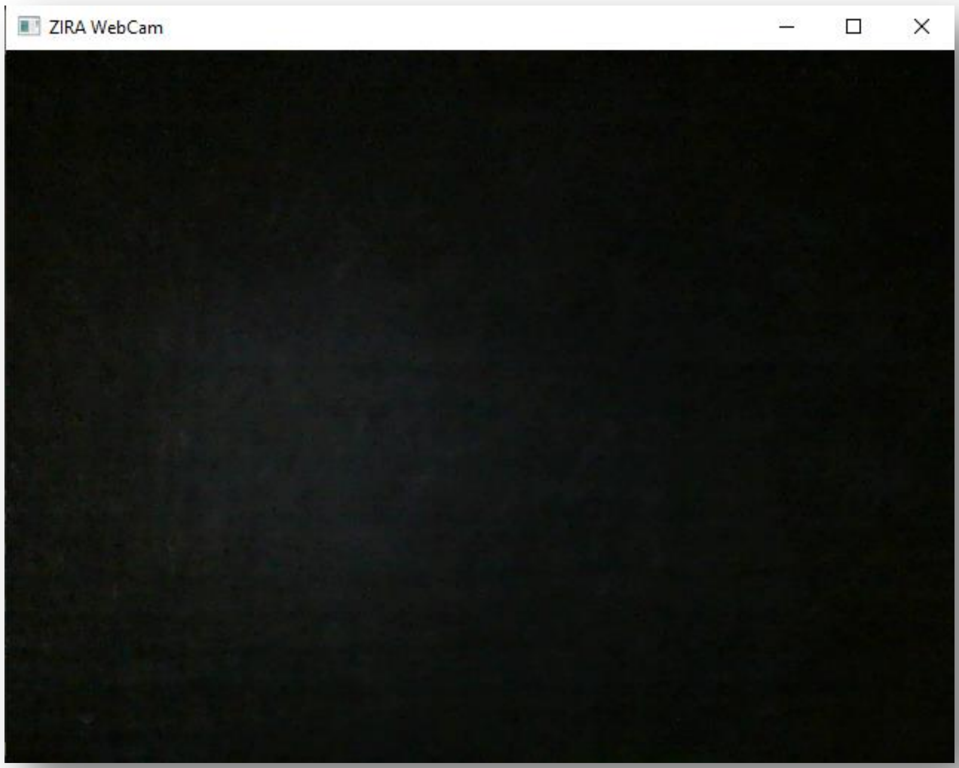
YOU: What is ai

Zira:

AI (Artificial Intelligence) is a branch of computer science that focuses on developing intelligent machines that can think and act like humans, and perform tasks such as problem-solving, decision-making, and natural language processing. AI technologies are used in a variety of fields, including robotics, healthcare, finance, and education.

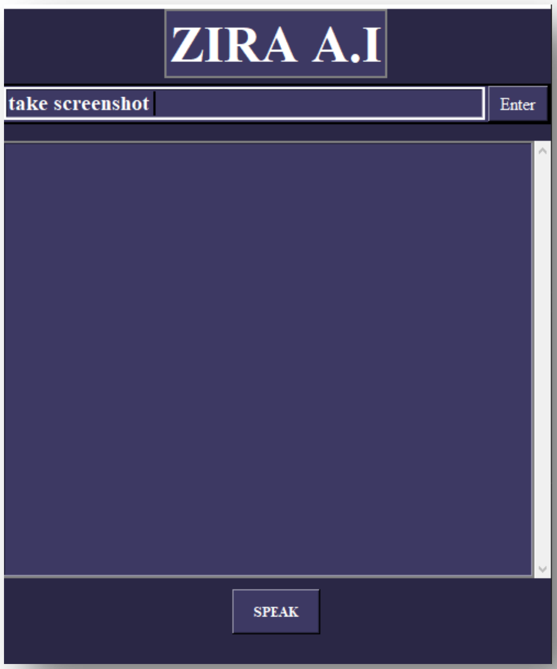
3.5.4 Answer shows

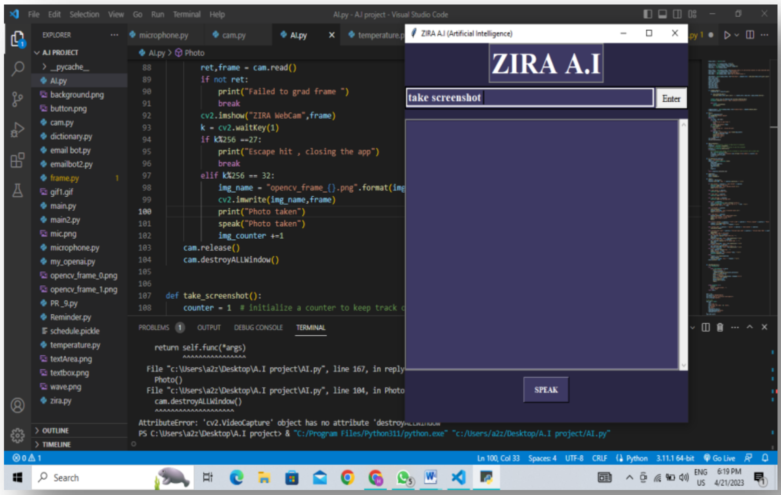
- The “Photo” function allow user to take Photo with the help of cv2 module. First a popup show that AI speak “Press Spacebar to take photo” and “Press Escape to exit”. If user press spacebar it take the photo and save the photo in the same window where program is running.



3.5.5 ZIRA Webcam

- The “screenshot option” allow user to the screenshot whenever user speak take screen-shot it take the screenshot then a pop up show to select the location to save the screen-shot.





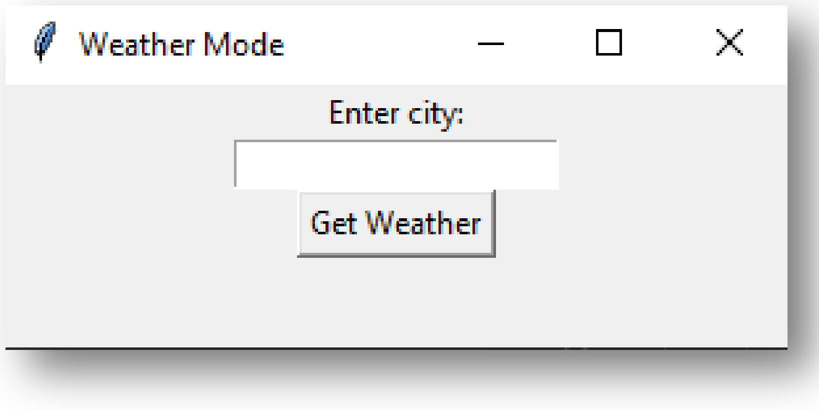
```
        ret,frame = cam.read()
        if not ret:
            print("Failed to grad frame ")
            break
        cv2.imshow("ZIRA WebCam",frame)
        k = cv2.waitKey(1)
        if k%256 ==27:
            print("Escape hit , closing the app")
            break
        elif k%256 == 32:
            img_name = "opencv_frame_{}.png".format(img
            cv2.imwrite(img_name,frame)
            print("Photo taken")
            speak("Photo taken")
            img_counter +=1
    cam.release()
    cam.destroyALLWindow()


def take_screenshot():
    counter = 1  # initialize a counter to keep track o
```

```
    return self.func(*args)
           ^^^^^^^^^^^^^^^^
  File "c:\Users\a2z\Desktop\A.I project\AI.py", line 167, in reply
    Photo()
  File "c:\Users\a2z\Desktop\A.I project\AI.py", line 104, in Photo
    cam.destroyALLWindow()
    ^^^^^^^^^^^^^^^^^^^^
AttributeError: 'cv2.VideoCapture' object has no attribute 'destroy
PS C:\Users\a2z\Desktop\A.I project> & "C:/Program Files/Python311/python.exe" "c:/Users/a2z/Desktop/A.I project/AI.py"
```

3.5.6 Screenshot

- The “weather option” allow user to check the weather of any city. It use “request” Module to request a website (OpenWeatherMap.com) to get the current weather of the city.



3.5.6 Weather GUI

- Overall, the user interface of the project is simple and easy to use, with clear instructions and buttons that are labeled with what they do. The design of the interface makes it easy for users to navigate and use the program's different functions without any difficulty.

3.6 Testing and Quality Assurance

- Testing and Quality Assurance are critical components of software development that ensure the software functions correctly and meets the needs of the end-users. In this project, the functions that require testing include email sending, scheduling, speech recognition, text-to-speech, web browsing, and screenshot-taking.

- To ensure that the email sending function works correctly, the email addresses of the recipients must be valid, and the subject and body text must be transmitted correctly. Unit tests can be written to check whether these inputs are being processed correctly. In addition, integration tests can be written to verify that the email is actually sent and received by the recipient.

- The scheduling function can be tested by verifying that emails are sent at the scheduled time. This can be done by running the function and checking that the email is received by the recipient at the specified time.

- The speech recognition and text-to-speech functions can be tested by providing different inputs and verifying that the output is accurate. This can be done by comparing the output with the expected output for each input.

- The web browsing function can be tested by checking that the correct webpage is opened in the browser. This can be done by using a testing framework such as Selenium to automate the process of opening the webpage and verifying that the correct URL has been loaded.

- The screenshot-taking function can be tested by verifying that the screenshot is captured and saved to the correct directory. This can be done by providing inputs and checking that the output matches the expected output.

- In conclusion, Testing and Quality Assurance are critical aspects of software development, and it is essential to test each function to ensure that the software works as intended. Proper testing ensures that the software is reliable and meets the needs of the end-users.

4.Implementation

4.1 Implementation:

- The GUI for the personal assistant is created using the Tkinter module. The main window contains a text widget for displaying the assistant's responses and a text entry field for user queries. The assistant's responses are also converted to speech output using the pyttsx3 module.

- The "reply" function handles the user's queries and generates a response using the OpenAI module. The assistant can also perform additional tasks, such as playing a song on YouTube, opening a web browser, or taking a screenshot.

- The "open_popup" function creates a pop-up window for scheduling an email to be sent later. The user can enter the recipient email addresses, subject, and message body. The "send_email" function sends the email using the smtplib module, and the "schedule_email" function schedules the email to be sent automatically at the specified time using the schedule module.

- The "Photo" function captures an image using the computer's webcam and saves it to a file using the OpenCV module. The "take_screenshot" function takes a screenshot of the user's screen using the pyautogui module.

4.2 HARDWARE AND SOFTWARE:

The software is designed to be light-weighted so that it doesn’t be a burden on
the machine running it. This system is being build keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirement for virtual assistant

1. **Hardware:**
   - Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30 GHz
   - RAM 512MB or more.

- 64-bit operating system, x64-based processor

2. **Software:**
   - Windows 10 Pro
   - Python 3.11.1
   - Chrome
   - Visual studio code

4.3 Feasibility Study

Feasibility study can help you determine whether or not you should proceed with your project. It is essential to evaluate cost and benefit. It is essential to evaluate cost and benefit of the proposed system. Five types of feasibility study are taken into consideration.

- **Technical feasibility:** It includes finding out technologies for the project, both hardware and software. For virtual assistant, user must have microphone to convey their message and a speaker to listen when system speaks. These are very cheap now a days and every-one generally, possess them. Besides, system needs internet connection. While using it, make Sure, you have a steady internet connection. It is an issue if we used without inter-net connection.

- **Operational feasibility:** It is the ease and simplicity of operation of proposed system. System does not require any special skill set for users to operate it. In fact, it is designed to be used by almost everyone. Kids who still don't know to write can readout problems for system and get answers.

- **Economic feasibility:** Here, we find the total cost and benefit of the Proposed system over current system. For this project, the main cost is documentation cost. User also would have to pay for microphone and speakers. Again, they are cheap and available. As far as maintenance is concerned, it won't cost too much.

- **Organizational feasibility:** This shows the management and organizational structure of the project. This project is not built by a team. The management tasks are all to be carried out by a single person. That won't create any management issues and will increase the feasibility of the project.

- **Cultural feasibility:** It deals with compatibility of the project with cultural environment. Virtual assistant is built in accordance with the general culture. This project is technically feasible with no external hardware requirements. also, it is simple in operation and does not cost training or repairs. Overall feasibility study of the project reveals that the goals of the proposed system is achievable. Decision is taken to proceed with the project.

5.Future Scope and Enhancement

5.1 Possible Enhancements:

- **Improve user interface**: The current user interface of the application can be enhanced to make it more visually appealing and user-friendly. The addition of icons, colors, and design elements can make it more attractive and engaging for the users.

- **Add support for more email providers:** Currently, the email sending functionality of the application only supports Gmail. To make the application more versatile, support for other email providers such as Yahoo, Outlook, etc. can be added.

- **Implement voice commands:** Currently, the application relies on user input through text fields. Implementing voice commands can make the application more accessible and convenient for users who prefer using voice commands.

- **Add more AI capabilities:** The application currently uses OpenAI's GPT-3 to generate responses to user questions. More advanced AI capabilities can be added to enhance the accuracy and relevance of the responses.

- **Integrate with other services**: The application can be enhanced by integrating with other popular services such as Spotify, Twitter, Facebook, etc. This can provide users with more options and increase the overall usefulness of the application.

- **Add support for multiple languages**: The application currently only supports English. Adding support for other languages can make it more accessible and useful for non-English speakers.

- **Implement more advanced computer vision features:** The current screen-shot functionality of the application is basic. Advanced computer vision features such as object detection, face recognition, etc. can be added to enhance the functionality of the application.

- **Improve the email scheduling functionality**: The email scheduling functionality can be improved by adding more options such as recurring emails, selecting specific days for sending emails, etc.

- **Implement automatic email response:** The application can be enhanced by implementing automatic email response functionality. This can be useful for users who are unable to respond to emails immediately.

- **Add more advanced text-to-speech features:** The current text-to-speech functionality of the application is basic. Advanced features such as changing the voice, speed, pitch, etc. can be added to enhance the functionality of the application.

5.2 Future Scope:

There is always room for improvement and expansion in any project, and the same is true for this project. Here are some areas where the project could be further developed:

- **Integration of more APIs:** The current project uses a limited number of APIs, and more APIs can be integrated to expand its functionality. For example, APIs can be used for weather forecasts, news updates, and stock prices.

- **GUI improvement:** The GUI can be further improved by making it more intuitive and user-friendly. The addition of more icons and buttons can help make it more visually appealing.

- **Multi-language support:** The current project is only available in English. Adding multi-language support can make it more accessible to a wider range of users.

- **Voice recognition improvement:** The accuracy of the voice recognition feature can be improved. This can be achieved by using more advanced machine learning models.

- **Integration with other applications:** The project can be integrated with other applications such as Google Drive, Dropbox, and Slack. This can help users to access their files and data easily.

- **Mobile application:** A mobile application can be developed for this project to make it more accessible to users who prefer using their smartphones.

- **Natural language processing improvement:** The natural language processing feature can be improved to provide better responses to user queries. This can be achieved by using more advanced machine learning models.

- **Automated testing:** Automated testing can be implemented to ensure the project's stability and reliability.

- **Integration with social media:** The project can be integrated with social media platforms such as Facebook, Twitter, and Instagram. This can help users to share their experience with others and increase the project's popularity.

- **Image and video processing:** The current project has limited image and video processing capabilities. Adding more features for image and video processing can expand its functionality. For example, it can be used for image and video editing or for face detection.

In conclusion, this project has a lot of potential for future development and expansion. With the integration of more APIs, improvement in the GUI and voice recognition, and adding more features, it can become a comprehensive solution for various tasks.

5.3 Limitations and Challenges of the Project:

The project presented here is a multi-purpose tool built using Python's tkinter library. The tool can perform various tasks such as sending emails, scheduling emails, taking screenshots, using OpenAI's GPT-3 API for question-answering, playing YouTube videos, opening websites, and taking pictures using a webcam. While the project is a great demonstration of the capabilities of Python and its libraries, it also has some limitations and challenges that need to be considered.

- **Limited functionality:** Although the tool offers various features, it is not exhaustive. For example, the tool could have included more advanced image processing capabilities, such as object detection or image recognition.

- **Dependence on external libraries:** The project relies on several external libraries, such as OpenAI, pyttsx3, pywhatkit, and pyautogui. While these libraries are well-maintained and widely used, there is a risk of compatibility issues with future releases.

- **Lack of error handling:** The project lacks sufficient error handling, which can cause the program to crash or behave unexpectedly in certain situations. For example, when sending emails, the program assumes that the user will enter valid email addresses and doesn't account for errors or exceptions.

- **Limited cross-platform compatibility:** The project is developed using Python, which is a cross-platform language. However, the GUI elements of the tool rely on the tkinter library, which may not be available or behave differently on all platforms.

- **Limitations of APIs:** The project relies on several APIs, such as OpenAI's GPT-3 API, which has limitations on the number of requests per month, and pyttsx3's text-to-speech API, which has limitations on the supported voices and languages.

- **Complexity:** The project is complex and may be challenging for beginners to understand and modify. The code could have been structured more effectively to improve readability and maintainability.

6. Testing

6.1 Testing:

To ensure the quality of the developed software, rigorous testing was carried out at each stage of the development process. Testing involved a combination of manual testing and automated testing.

Unit testing was performed during the coding phase to ensure that each module of the software was functioning as intended. Integration testing was carried out during the integration phase to ensure that the different modules of the software were working well together.

In addition to these, functional testing was carried out to ensure that the software met the functional requirements specified in the project scope. This involved testing the various features of the software to ensure that they were working as expected. Non-functional testing was also performed to ensure that the software was efficient, reliable, and secure.

To ensure that the software was usable and user-friendly, usability testing was carried out with a group of users who were representative of the target audience. The feedback obtained from these users was used to improve the user interface and overall usability of the software.

Finally, acceptance testing was carried out to ensure that the software met the expectations of the stakeholders. This involved demonstrating the software to the stakeholders and obtaining their approval before releasing the software for production use.

Overall, testing played a critical role in ensuring that the software met the functional and non-functional requirements specified in the project scope and that it was of high quality.

6.2 Testing Principles:

In software development, testing is a critical phase to ensure the quality of the product. The following are some of the fundamental testing principles that can guide the testing process:

1. Exhaustive Testing is impossible:

It is not possible to test every possible input, output, or scenario in the software. Hence, testing should be focused on high-risk areas, critical functionalities, and common usage scenarios.

2. Testing should start early:

Testing should start early in the software development life cycle (SDLC) to identify and address issues as early as possible. This helps to minimize the cost and effort required to fix defects.

3. Defect clustering:

In most software applications, a small number of modules or functionalities contain most of the defects. Identifying and addressing these high-risk areas can help to improve the overall quality of the software.

4. Pesticide paradox:

If the same set of test cases is repeatedly executed, eventually, they become ineffective in finding new defects. Hence, test cases should be reviewed and updated regularly to ensure that they are effective in identifying defects.

5. Testing is context-dependent:

The testing approach and strategy should be tailored to the context of the software project. The context includes factors such as the software domain, development methodology, project constraints, and team skills.

6. Absence of errors fallacy:
The absence of errors in the software does not imply that the software is defect-free. It is always possible that some defects are not discovered during testing. Hence, testing should be complemented with other quality assurance activities such as code reviews, static analysis, and dynamic analysis.

7. Testing should be objective:
Testing should be conducted with a clear objective to identify defects and improve the quality of the software. Testers should avoid being biased towards finding or not finding defects based on their assumptions, beliefs, or prior knowledge.

By following these testing principles, the testing process can be made more effective, efficient, and focused on improving the quality of the software.

6.3 Testability:

Testability is an important aspect of software development that ensures that the software is designed and developed in a way that makes it easy to test. In other words, it is the ease with which a software system or component can be tested. The more testable a software system is, the easier it is to detect and fix bugs.

In this project, testability was a key consideration throughout the development process. The following are some of the ways in which the project team ensured testability:

1. Modular Design: The software was designed to be modular, with each module performing a specific function. This made it easier to isolate and test each module separately, which in turn made it easier to identify and fix issues.

2. Code Review: All code changes were reviewed by at least two members of the team, which helped ensure that the code was well-structured, easy to understand, and easy to test.

2. **Use of Mocks and Stubs:** The project team used mocks and stubs to simulate external dependencies and ensure that the software worked as expected in various scenarios.

Overall, the project team focused on making the software as testable as possible, which helped to ensure that it was of high quality and met the requirements of the client.

7. Conclusion

7.1 Conclusion:

The AI personal assistant with email scheduler and other features is a useful tool for automating several tasks and making daily life easier. The assistant can perform a wide range of tasks, including sending emails, browsing the web, and generating responses to user queries using the OpenAI module. The assistant can also capture images and take screenshots, making it a useful tool for both personal and professional use.

7.2 Summary of the project:

- The project is a desktop assistant that performs various tasks such as sending emails, taking screenshots, answering questions, and much more. The project has been developed using the Python programming language and the Tkinter module for the graphical user interface.

- The assistant can send emails to multiple recipients at a scheduled time using the SMTP protocol. The user can enter the recipient's email addresses, the subject of the email, the body of the email, and the time when the email should be sent. The assistant uses the Schedule module to schedule the email and sends the email at the specified time.

- The assistant can take screenshots using the PyAutoGUI module. The user can select the directory to save the screenshot, and the assistant saves the screenshot with a unique name.

- The assistant can also answer questions using the OpenAI API. The user can ask any question, and the assistant will use the text-davinci-003 model to provide the answer.

- The assistant can also open websites such as YouTube, Google, and Yahoo using the webbrowser module.

- Finally, the assistant can recognize speech and respond to the user using the pyttsx3 module. The user can speak a command, and the assistant will execute the command accordingly. The assistant can also take pictures using the webcam of the system using the OpenCV module.
- Overall, the desktop assistant project provides a comprehensive solution for automating tasks and making the user's life easier.

7.3 Future Recommendations:

- Enhancing the AI capabilities: Currently, the chatbot uses OpenAI's text-davinci-003 model to generate responses to user questions. In the future, the model can be upgraded to a more advanced model that can handle complex and contextual conversations with greater accuracy.

- Adding a GUI: The current application is a command-line interface. To make the application more user-friendly, a graphical user interface (GUI) can be developed using the Tkinter library, which can improve the application's usability and aesthetics.

- Improving email scheduling: The email scheduling feature can be improved by adding the ability to select the frequency of email sending, such as daily, weekly, or monthly. Additionally, a calendar widget can be added to the GUI to make it easier for users to select the date and time.

- Voice recognition: Voice recognition capabilities can be added to the chatbot to allow users to interact with the application using voice commands. The speech_recognition library can be used to achieve this.

- Adding more features: The chatbot can be enhanced with additional features such as weather forecasting, news updates, and stock market tracking, among others. This will provide users with a more comprehensive and valuable experience.

8. References

8.1 References:

1. https://openai.com/
2. https://openweathermap.org/
3. https://mermaid.live/
4. https://www.tutorialspoint.com/python/index.htm
5. https://docs.python.org/3/tutorial/

8.2 YouTube References:

1. https://www.youtube.com/@CodeWithHarry
2. https://www.youtube.com/@wscubetech
3. https://www.youtube.com/@codinglifestyle4u
4. https://youtu.be/IhRfqiC29Ds