**A**

**PROJECT REPORT**

**ON**

**S.A.G.E (SMART ASSISTANT FOR GREATER EFFICIENCY)**

BACHELOR’S OF SCIENCE

IN

COMPUTER SCIENCE

**SUBMITTED BY**

**SHAIKH SHAHID**



**S.K. COLLEGE OF SCIENCE AND COMMERCE,**

**PLOT NO.31, SEC.25, SEAWOODS , NAVI MUMBAI – 400706**

**2024-25**

**A**

**PROJECT REPORT**

**ON**

**S.A.G.E (SMART ASSISTANT FOR GREATER EFFICIENCY)**

BACHELOR’S OF SCIENCE

IN

COMPUTER SCIENCE

**SUBMITTED BY**

**SHAIKH SHAHID**

PROJECT GUIDE

DR. SHRADDHA SABLE

**S.K. COLLEGE OF SCIENCE AND COMMERCE,**

**PLOT NO.31, SEC.25, SEAWOODS , NAVI MUMBAI - 400706**

**2024-25**

**ABSTRACT**

**S.A.G.E. (Smart Assistant for Greater Efficiency)** is a sophisticated desktop-based voice assistant developed to significantly enhance user productivity by automating and managing a wide range of tasks. Powered by TensorFlow, S.A.G.E. offers accurate voice command recognition, enabling seamless control over applications, file management, and system operations. The assistant also supports communication through WhatsApp, facilitating tasks like sending messages and making voice or video calls. In addition to these core functions, S.A.G.E. retrieves real-time news and weather updates, offers entertainment recommendations, and includes interactive features such as games, reminders, and alarms. With robust multimedia capabilities, including taking photos, screenshots, and managing screen recordings, S.A.G.E. serves as a versatile tool for both personal and professional use. Its secure, user-specific access ensures that all interactions and data are managed safely, making it an indispensable asset for streamlining digital workflows and enhancing overall efficiency.

**ACKNOWLEDGEMENT**

I would like to express my special thanks of gratitude to my project guide **Dr. Shraddha Sable** for their guidelines and support in making my project successful.

I am very thankful to our principal **Dr. Swati Vitkar** for their kind cooperation in completion of my project.

I am also grateful to **Mrs. Vrushali Ghatpande**, head of department for being very much resourceful, kind and helpful. And also thankful for their positive attitude, time and efforts she provided to us. Your useful advices gives us confidence to do project successfully.

I also wish to thanks our lab assistant and all my friends and entire **computer department** who helped me in completion of my project.

Last but not least I would like to thank my family without whose support , motivation and encouragement it would not have been possible.

**PREFACE**

It gives me great pleasure to present the project on “**S.A.G.E (SMART ASSISTANT FOR GREATER EFFICIENCY)**” prepared sincerely, punctually and with utmost efforts.

The contents of this book are presented into many chapters, in order to ease reading. The project has been illustrated with precise data elucidated with neat Sequence Diagram and Activity Diagram that simplify the understanding of the project. The project includes the software development tools like Visual Studio Code and the integration of various APIs.

Meticulous care has been taken to make this project perfect and useful in every respect.

**SHAIKH SHAHID**

**DECLARATION**

I the undersigned **Mr. SHAIKH SHAHID** hereby, declare that the work embodied in this project work titled **“S.A.G.E (SMART ASSISTANT FOR GREATER EFFICIENCY)”** forms my own contribution to the research work carried out under the guidance of **DR. SHRADDHA SABLE** is a result of my own research work and has not been previously submitted to any other University for any other Degree/ Diploma to this or any other University.

Wherever reference has been made to previous works of others, it has been clearly indicated as such and included in the bibliography.

I will abide and take all responsibility of all issues related to plagiarism/self-plagiarism mentioned under UNIVERSITY GRANTS COMMISSION (PROMOTION OF ACADEMIC INTEGRITY AND PREVENTION OF PLAGARISM IN HIGHER EDUCATIONAL INSTITUTIONS) REGULATIONS, 2021.

I, here by further declare that all information of this document has been obtained and presented in accordance with academic rules and ethical conduct.

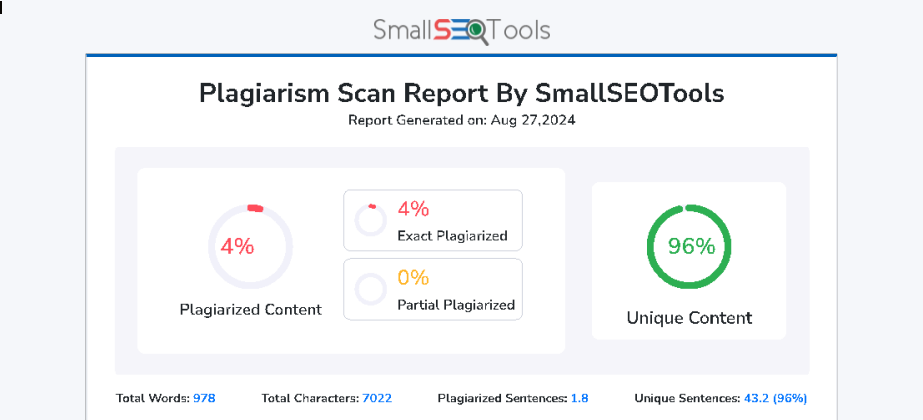
Submitted by Certified by

**Mr. Shaikh Shahid**  **Dr. Shraddha Sable**

**INDEX**

|  |  |
| --- | --- |
| INDEX | TOPIC |
| 1 | **PRELIMINARY INVERTIGATION** |
|  | i. INTRODUCTION |
|  | ii. OBJECTIVE |
| 2 | **SCOPE OF THE PROJECT** |
|  | i. SCOPE |
|  | ii. PURPOSE |
|  | iii. APPLICABILITY |
| 3 | **REQUIREMENT SPECIFICATION** |
|  | 1. SOFTWARE REQUIREMENTS |
|  | ii. HARDWARE REUIREMENRTS |
|  | iii. FUNCTIONAL REQUIREMENTS |
| 4 | **SYSTEM ANALYSIS** |
|  | 1. ACTIVITY DIAGRAM |
|  | 1. SEQUENCE DIAGRAM |
|  | iii. DATA FLOW DIAGRAM |
| 5 | **IMLEMENTATION & DESIGN** |
|  | 1. CODE |
|  | 1. DESIGN |
| 6 | **FUTURE ENHANCEMENT** |
| 7 | **SUMMARY & CONCLUSION** |
|  | i. SUMMARY |
|  | ii. CONCLUSION |
| 8 | **REFERENCES AND BIBLIOGRAPHY** |

**PLAGIARISM REPORTS**





1. **PRELIMINARY INVERTIGATION**

**INTRODUCTION**

Automation is rapidly becoming an integral part of our daily lives, as machines are increasingly capable of performing tasks autonomously with the help of cutting-edge technologies like Machine Learning (ML) and Neural Networks. Among the most prominent examples of these technologies in action are Virtual Assistants. These programs enable us to interact with our devices using natural language commands, typically through voice recognition, thereby simplifying and streamlining various tasks.

Virtual Assistants have revolutionized how we engage with technology. A simple voice command such as "What's the weather today?" can trigger the assistant to provide you with the latest weather report specific to your location. This seamless interaction is made possible by the assistant's ability to understand and process natural language, thanks to advancements in ML and Natural Language Processing (NLP).

Inspired by these advancements, I embarked on a journey to create my own Voice Assistant, named "S.A.G.E" (Smart Assistant for Greater Efficiency). S.A.G.E. is designed to function as a versatile desktop assistant, capable of assisting users with a wide range of tasks. From providing real-time news and weather updates to managing communications, such as sending messages and making calls, to performing web searches, S.A.G.E. is designed to be a comprehensive tool that enhances both personal and professional productivity.

**OBJECTIVE**

The primary objective of this project is to develop **S.A.G.E. (Smart Assistant for Greater Efficiency)**, a highly efficient and versatile AI-powered voice assistant designed to significantly enhance user productivity by automating a broad spectrum of tasks. S.A.G.E. is engineered to seamlessly integrate a wide array of functionalities, including application control, file management, setting alarms, and reminders. It also offers interactive features like games, making it both a useful and engaging tool for users.

A key strength of S.A.G.E. is its ability to provide real-time updates on news, weather, and other relevant information, making it a comprehensive information hub. By leveraging advanced machine learning models and natural language processing techniques, particularly using TensorFlow for accurate intent recognition, S.A.G.E. ensures precise execution of user commands, enhancing the overall user experience. The voice assistant is designed to handle complex queries and multitask effectively, allowing users to perform various functions simultaneously, such as opening multiple applications or browsing the web while managing their daily schedules.

S.A.G.E. is built with a focus on simplicity and accessibility, utilizing a combination of Python, JavaScript, HTML, CSS, and Eel to create a modern, user-friendly interface. The intuitive design allows users to perform tasks using voice commands, reducing the need for manual input and making the assistant suitable for a wide range of users, including those with limited technical skills.

**2. SCOPE OF THE PROJECT**

**SCOPE**

The scope of the S.A.G.E. project encompasses the development of a comprehensive voice assistant that integrates a multitude of functionalities. These functionalities include:

* **Application Control:** S.A.G.E. can open, close, and manage various desktop applications based on voice commands, providing hands-free control over the user's workspace.
* **File Management:** Users can create, delete, and organize files and folders through simple voice commands, making file management more intuitive and efficient.
* **System Operations:** S.A.G.E. is capable of managing essential system functions, such as window control, volume adjustment, and executing system commands like shutdown or restart.
* **Interactive Features:** The assistant includes entertainment options such as games, and utility features like reminders, alarms, and timers to help manage daily activities.
* **Communication Tools:** S.A.G.E. facilitates communication by enabling the sending of messages via WhatsApp, as well as making voice and video calls, integrating seamlessly with the user’s communication tools.
* **Web Interaction:** The assistant can open websites and perform searches on platforms like Google and YouTube, making information retrieval faster and more efficient.
* **Information Retrieval:** S.A.G.E. provides real-time updates on news and weather, and offers recommendations for movies or TV shows, catering to the user's information and entertainment needs.
* **Multimedia Capabilities:** The assistant can take photos, capture screenshots, and start or stop screen recordings, adding a visual and interactive dimension to its functionality.

The project is thoroughly documented and tested to ensure reliability and performance. S.A.G.E. is designed with scalability in mind, allowing for easy integration of future features as technology and user needs evolve.

**PURPOSE**

The purpose of S.A.G.E. is to deliver a highly functional and all-encompassing voice assistant that streamlines a wide range of daily activities, thereby enhancing user productivity. By integrating an extensive range of functionalities—ranging from basic application management to advanced system control, communication tools, and real-time information updates—S.A.G.E. is intended to be a one-stop solution for managing both personal and professional tasks. The assistant is designed to make technology more accessible and user-friendly, automating repetitive tasks and simplifying complex operations through intuitive voice commands.

S.A.G.E. aims to redefine the way users interact with their digital workspace by offering a seamless and efficient interface. Whether it’s managing files, controlling system functions, or retrieving information, the assistant ensures that users can focus more on their core activities while S.A.G.E. handles the rest. By doing so, it not only enhances productivity but also contributes to a more organized and stress-free digital environment.

**APPLICABILITY**

S.A.G.E. is applicable in a wide range of settings, offering unparalleled convenience, automation, and efficiency for both personal and professional use. Its versatile nature makes it an ideal tool for managing daily tasks, streamlining workflows, facilitating communication, and providing entertainment and information. The assistant is designed to cater to a diverse audience, making it suitable for students, professionals, and even casual users who wish to simplify their interaction with digital tools.

In a professional setting, S.A.G.E. can serve as a productivity booster by automating routine tasks, such as managing applications, organizing files, and retrieving information, thereby allowing users to focus on more critical work. In a personal context, it offers a variety of features that enhance day-to-day activities, from setting reminders and alarms to providing news updates and entertainment options makes S.A.G.E. a highly applicable and valuable tool in today’s digital age.

**3. REQUIREMENT ANALYSIS**

**SOFTWARE REQUIREMENTS**

* **Operating System:** Windows 10 or higher.
* **Python 3.x:** For running the S.A.G.E. application and scripts.
* **TensorFlow:** Used for natural language processing and identifying user intent.
* **Eel:** For the graphical user interface.
* **Pycaw:** For system audio control.
* **Pywinauto:** For automating interactions with other applications.
* **Additional Libraries:** Other Python libraries such as OpenCV (for camera functions), Requests (for API calls), and more as needed.

**HARDWARE REUIREMENRTS**

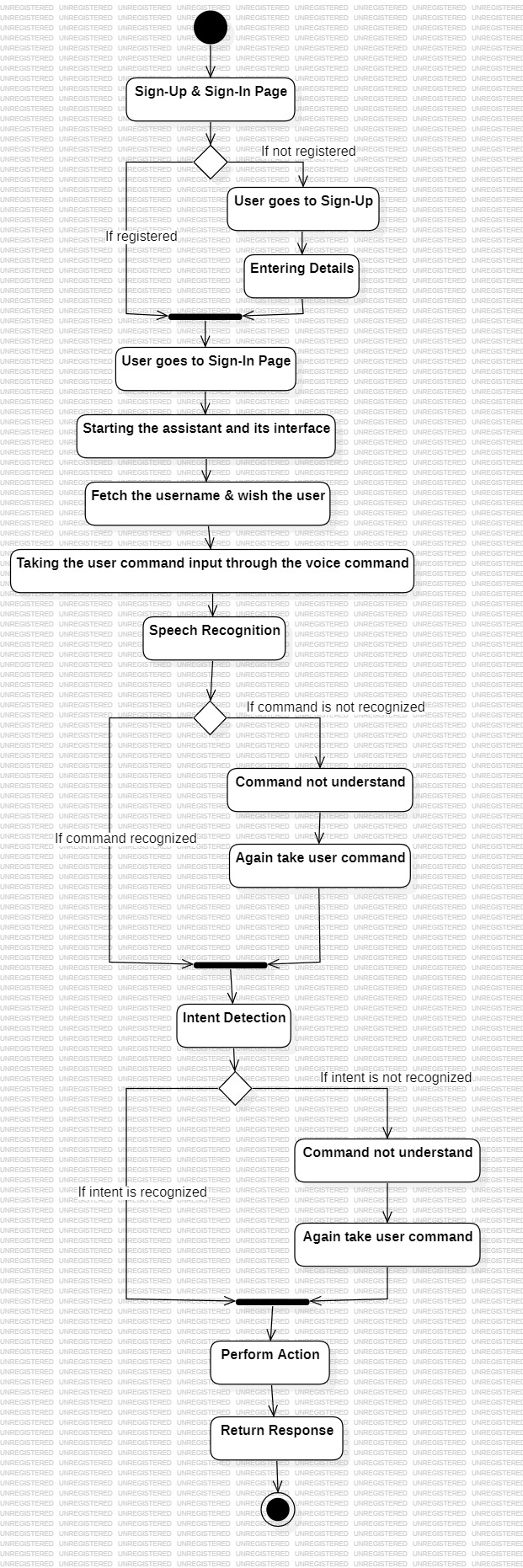
* **Processor:** A multi-core processor, such as Intel i5 or equivalent, to effectively manage multiple tasks at once.
* **Memory:** A minimum of 4 GB of RAM to ensure seamless multitasking and smooth operation.
* **Storage:** A minimum of 100 GB of hard drive space for storing data, recordings, and other relevant files.
* **Microphone:** A high-quality microphone for accurate voice command recognition.
* **Camera:** An integrated or external camera for capturing photos and videos as part of the assistant's multimedia functionality.

**FUNCTIONAL REQUIREMENTS**

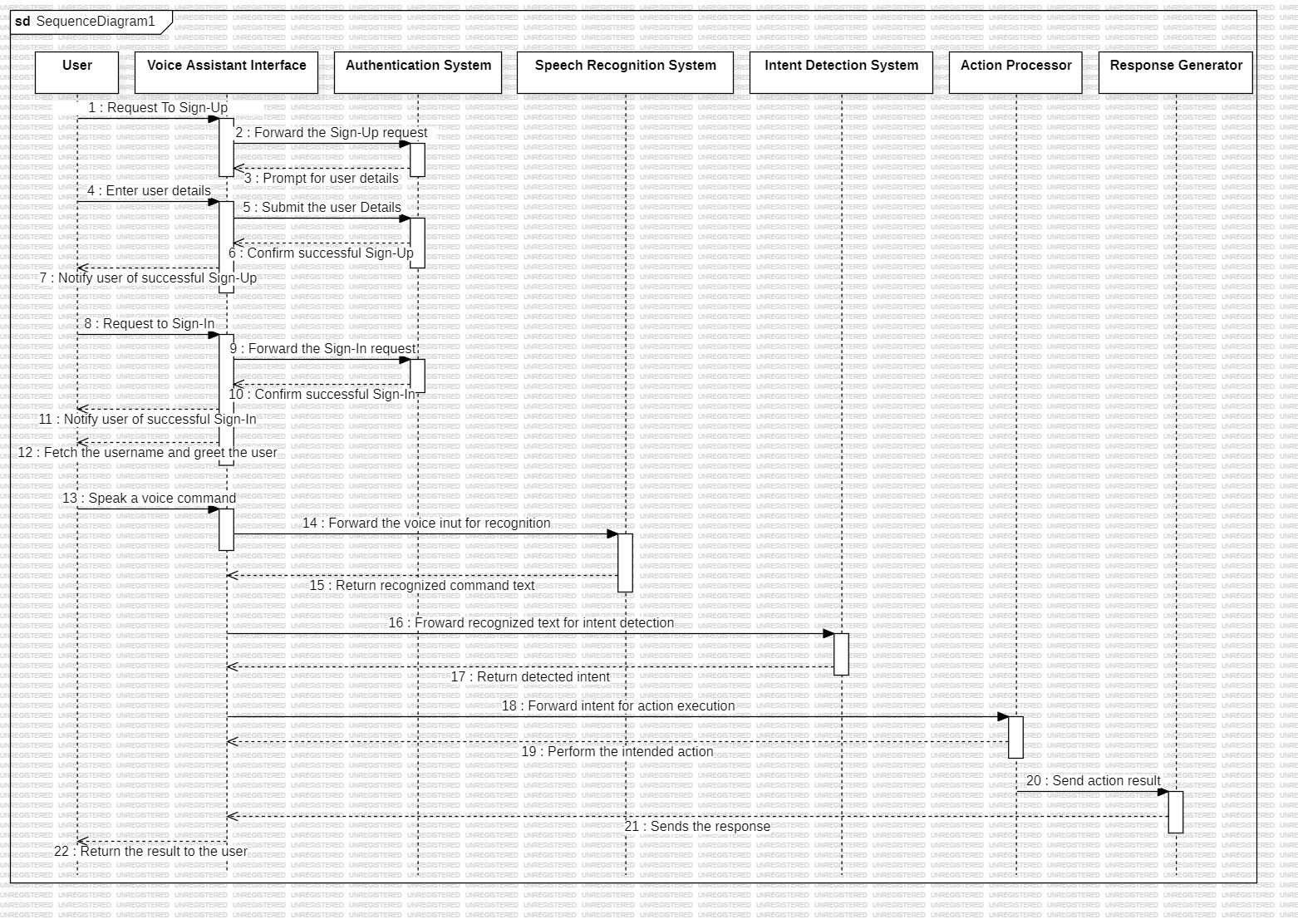
* **Voice Command Recognition:** Accurate recognition and response to diverse voice commands using TensorFlow.
* **Application Control:** Open, close, and manage applications via voice.
* **File Management:** Create, delete, and organize files and folders.
* **System Operations:** Control windows, tabs, volume, and execute system commands like shutdown or restart.
* **Communication:** Send WhatsApp messages and make voice/video calls.
* **Web Interaction:** Open websites and search on Google and YouTube.
* **Information Retrieval:** Provide real-time news and weather updates.
* **Interactive Features:** Play games, set reminders, alarms, timers, and take voice notes.
* **Multimedia Functions:** Capture photos, screenshots, and start/stop screen recordings.Application Control: The ability to open, close, and manage various applications through voice commands.

**4. SYSTEM ANALYSIS**

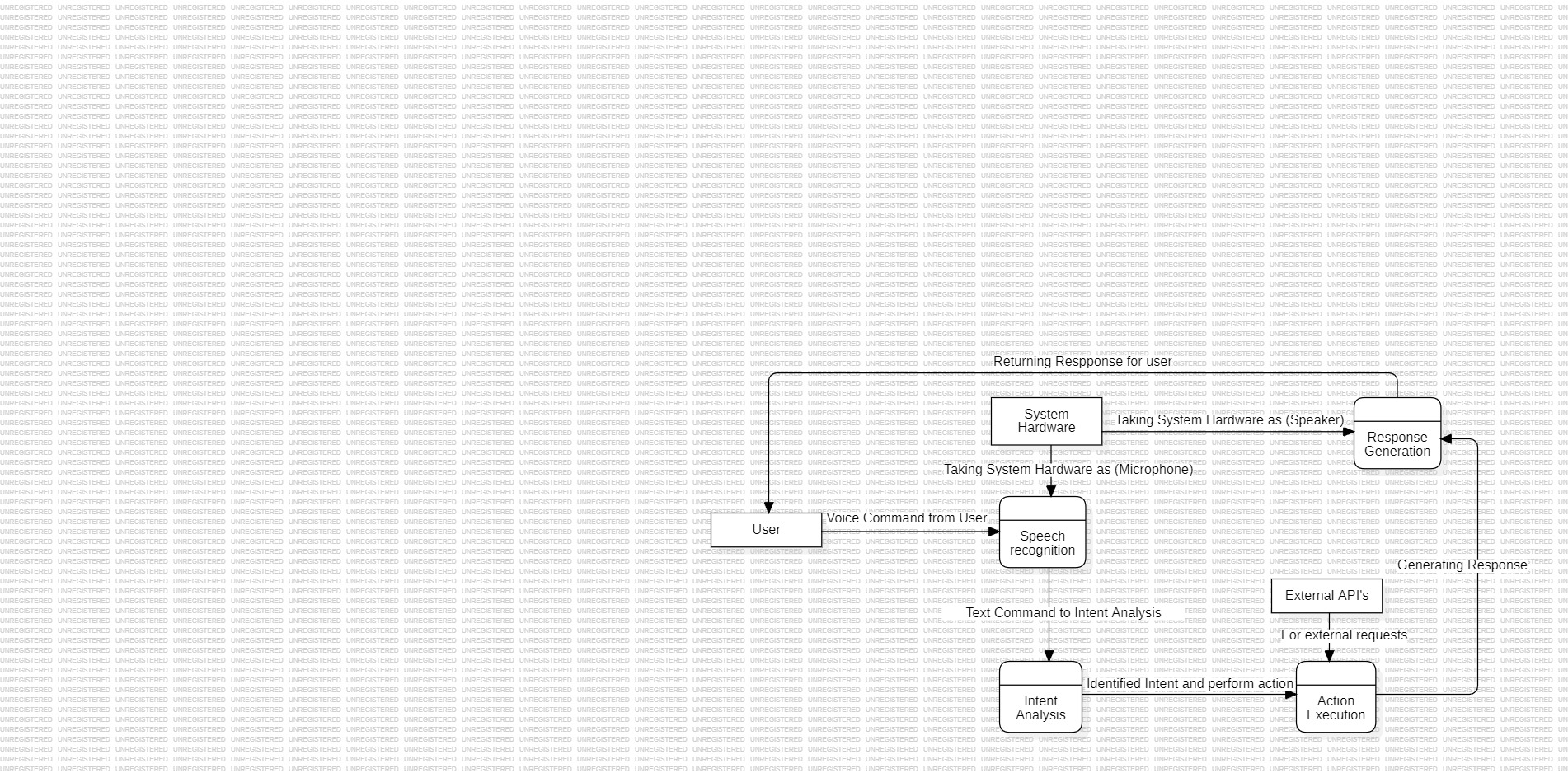
**- Activity Diagram**

****

**- Sequence Diagram**

****

**- Data Flow Diagram**

****

**5. IMPLEMENTATION & DESIGN**

**• CODE:**

from datetime import datetime

import pyttsx3

import speech\_recognition as sr

import eel

import sqlite3

import numpy as np

import tensorflow as tf

from tensorflow.keras.preprocessing.sequence import pad\_sequences

import pickle

import random

import re

# Importing command modules

from Commands.AppTask import \*

from Commands.FileOperations import \*

from Commands.Games import \*

from Commands.General import \*

from Commands.SystemTask import \*

from Commands.Websites import \*

from Commands.Whatsapp import \*

from Commands.Work import \*

# Initialize the text-to-speech engine

engine = pyttsx3.init('sapi5')

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[0].id)  # Select the voice (index 0 is typically the default)

engine.setProperty('rate', 200) # Set the speech rate

# Connect to the SQLite database

conn = sqlite3.connect('user\_data.db')

cursor = conn.cursor()

# Create users table if not exists

cursor.execute('''CREATE TABLE IF NOT EXISTS users (

                    id INTEGER PRIMARY KEY,

                    email TEXT UNIQUE,

                    username TEXT UNIQUE,

                    password TEXT

                )''')

conn.commit()

# Initialize Eel

eel.init('web')

# Global variables for hold and timer

on\_hold = True

last\_interaction\_time = datetime.now()

hold\_threshold = 10  # seconds

logged\_in\_username = None

def reset\_timer():

    global last\_interaction\_time

    last\_interaction\_time = datetime.now()

# Function to validate email addresses

def is\_valid\_gmail(email):

    """Check if the provided email is a valid Gmail address."""

    # Regex pattern for Gmail validation

    pattern = r'^[a-zA-Z0-9.\_%+-]+@gmail\.com$'

    if re.match(pattern, email):

        return True

    else:

        return False

@eel.expose

def registeruser(email, username, password):

    """Register a new user, allowing only @gmail.com addresses."""

    if not is\_valid\_gmail(email):

        speak("Please use a valid Gmail address to register.")

        return False

    try:

        cursor.execute('INSERT INTO users (email, username, password) VALUES (?,?,?)',

                       (email, username, password))

        conn.commit()

        return True

    except sqlite3.IntegrityError:

        # Handle duplicate username or email

        return False

@eel.expose

def loginuser(username, password):

    """Login an existing user"""

    cursor.execute("SELECT \* FROM users WHERE username=? AND password=?", (username, password))

    user = cursor.fetchone()

    if user:

        global logged\_in\_username

        logged\_in\_username = username

        return True

    else:

        return False

@eel.expose

def logoutuser():

    """Logout the current user and navigate to the login page"""

    global logged\_in\_username

    logged\_in\_username = None

def speak(audio):

    engine.say(audio)

    engine.runAndWait()

def get\_user\_name(username):

    """Retrieve the name of the logged-in user from the database."""

    cursor.execute("SELECT username FROM users WHERE username=?", (username,))

    result = cursor.fetchone()

    if result:

        return result[0]

    else:

        return None

def WishMe():

    hour = datetime.now().hour

    if 0 <= hour < 12:

        speak(f"Good Morning! {logged\_in\_username}")

    elif 12 <= hour < 16:

        speak(f"Good Afternoon! {logged\_in\_username}")

    else:

        speak(f"Good Evening! {logged\_in\_username}")

    assname = "Sage 1.0"

    if logged\_in\_username:

        speak(f", I am a Voice Assistant {assname}, and I am here to listen and complete your work with your voice commands.")

def takeCommand():

    r = sr.Recognizer()

    with sr.Microphone() as source:

        speak("Listening...")

        print("Listening...")

        r.pause\_threshold = 0.5

        audio = r.listen(source, phrase\_time\_limit=5)

    try:

        print("Recognizing...")

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

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

    except Exception as e:

        print("....")

        return "None"

    return query

def wake\_word(word="Sage"):

    r = sr.Recognizer()

    with sr.Microphone() as source:

        print("..........")

        audio = r.listen(source)

    try:

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

        if word.lower() in query.lower():

            print(f"Wake word detected: {query}")

            return True

        else:

            print(f"Wake word not detected: {query}")

            return False

    except Exception as e:

        return False

# Load the trained model

model\_path = 'my\_model.keras'

model = tf.keras.models.load\_model(model\_path)

# Load the tokenizer

tokenizer\_path = 'tokenizer.pickle'

with open(tokenizer\_path, 'rb') as handle:

    tokenizer = pickle.load(handle)

# Load the label encoder

label\_encoder\_path = 'label\_encoder.pickle'

with open(label\_encoder\_path, 'rb') as ecn\_file:

    label\_encoder = pickle.load(ecn\_file)

# Function to preprocess and predict intent

def predict\_intent(query):

    max\_len = 30

    sequence = tokenizer.texts\_to\_sequences([query])

    padded\_sequence = pad\_sequences(sequence, truncating='post', maxlen=max\_len)

    prediction = model.predict(padded\_sequence)

    predicted\_label = label\_encoder.inverse\_transform([np.argmax(prediction)])

    return predicted\_label[0]

@eel.expose

def start\_assistant():

    global on\_hold

    WishMe()

    on\_hold = False

    while True:

        if on\_hold:

            if wake\_word():

                speak("I am Listening, sir!")

                reset\_timer()

                on\_hold = False

        else:

            query = takeCommand().lower()

            intent = predict\_intent(query)

            if query == "none":

                if (datetime.now() - last\_interaction\_time).seconds > hold\_threshold:

                    on\_hold = True

                continue

            reset\_timer()

            # Process commands based on the detected intent

            if intent == "greeting":

                speak("Hello sir, How can i hel you today?")

            elif "what is my name" in query:

                if logged\_in\_username:

                    user\_name = get\_user\_name(logged\_in\_username)

                    if user\_name:

                        speak(f"Your name is {user\_name}.")

                    else:

                        speak("I'm sorry, I could not retrieve your name.")

                else:

                    speak("You are not logged in, so I don't know your name.")

            elif "who are you" in query.lower():

                speak('My name is Sage. I can perform a variety of tasks as programmed by my creator.')

            elif "who created you" in query.lower():

                speak('I am created by an anonymous coder to perform a variety of tasks.')

            elif "whats your full form" in query.lower() or "your full form" in query.lower():

                speak('Smart Assistant for Greater Efficiency')

            elif intent == "open\_application":

                app\_name = query.split("open", 1)[1].strip()

                speak(f"Opening {app\_name}")

                open\_application(app\_name)

            elif intent == "close\_application":

                speak("Closing the current application")

                close\_application()

            elif intent == "create\_file":

                F1 = ["What should be the name of the file?",

                            "Please tell the file name.",

                            "What's the file name?"]

                speak(random.choice(F1))

                file\_name = takeCommand().strip()

                F2 = ["What's the content of a file?",

                        "Please provide the content."]

                speak(random.choice(F2))

                content = takeCommand().strip()

                create\_file(file\_name, content)

            elif intent == "delete\_file":

                F3 = ["What is the name of a file to delete?",

                            "Please tell the file name.",

                            "What's the file name?"]

                speak(random.choice(F3))

                file\_name = takeCommand().strip()

                delete\_file(file\_name)

            elif intent == "create\_folder":

                F4 = ["What should be the name of a folder?",

                            "Please tell the folder name.",

                            "What's the folder name?"]

                speak(random.choice(F4))

                folder\_name = takeCommand().strip()

                create\_folder(folder\_name)

            elif intent == "delete\_folder":

                F5 = ["What is the name of the folder to delete?",

                            "Please tell the folder name.",

                            "What's the folder name?"]

                speak(random.choice(F5))

                folder\_name = takeCommand().strip()

                delete\_folder(folder\_name)

            elif intent == "rename\_item":

                speak("What is the current name of the item?")

                source\_name = takeCommand().strip()

                speak("What should be the new name of the item?")

                new\_name = takeCommand().strip()

                rename\_item(source\_name, new\_name)

            elif intent == "start\_game":

                start\_game()

            elif intent == "get\_news":

                G1 = ["Wait a minute sir.",

                      "One minute",

                      "Okay sir, wait a moment..",

                      "Just a moment please.",

                      "Give me a second to fetch that.",

                      "I'll have the news for you shortly."]

                speak(random.choice(G1))

                get\_news()

            elif intent == "get\_weather":

                G2 = ["Getting the weather report...",

                      "Hold on a minute..",

                      "Fetching the latest weather update...",

                      "Please wait a moment while I get the weather details.",

                      "Just a second, getting the weather info.",

                      "I'll have the weather report for you shortly."]

                speak(random.choice(G2))

                get\_weather()

            elif intent == "tell\_joke":

                tell\_joke()

            elif intent == "set\_reminder":

                set\_reminder(query.replace("remind me", "").strip())

            elif intent == "get\_reminders":

                get\_reminders()

            elif intent == "set\_timer":

                set\_timer()

            elif intent == "lookup\_word":

                lookup\_word(query.replace("define", "").strip())

            elif intent == "get\_random\_funfact":

                get\_random\_funfact()

            elif intent == "volume\_up":

                volume\_up()

            elif intent == "volume\_down":

                volume\_down()

            elif intent == "mute\_volume":

                mute\_volume()

            elif intent == "unmute\_volume":

                unmute\_volume()

            elif intent == "normalize\_volume" in query:

                normalize\_volume()

            elif intent == "switch\_tab" in query:

                switch\_tab()

            elif intent == "new\_tab" in query:

                new\_tab()

            elif intent == "close\_tab" in query:

                close\_tab()

            elif intent == "switch\_window" in query:

                switch\_window()

            elif intent == "minimize\_window":

                minimize\_window()

            elif intent == "maximize\_window":

                maximize\_window()

            elif intent == "take\_note" in query:

                take\_note()

            elif intent == "select\_all" in query:

                select\_all()

            elif intent == "delete\_text" in query:

                delete\_text()

            elif intent == "copy\_text" in query:

                copy\_text()

            elif intent == "cut\_text" in query:

                cut\_text()

            elif intent == "paste\_text" in query:

                paste\_text()

            elif intent == "undo" in query:

                undo()

            elif intent == "redo" in query:

                redo()

            elif intent == "swipe\_up" in query:

                swipe\_up()

            elif intent == "swipe\_down":

                swipe\_down()

            elif intent == "turn\_off\_screen":

                turn\_off\_screen()

            elif intent == "shutdown\_system":

                shutdown\_system()

            elif intent == "restart\_system":

                restart\_system()

            elif intent == "open\_website":

                website\_name = query.lower().replace("open website", "").strip()

                open\_website(website\_name)

            elif intent == "search\_google":

                G4 = ["What should I search on Google?",

                      "Please tell me what to search on Google.",

                      "What do you want to search for on Google?",

                      "Can you specify what to search on Google?",

                      "What would you like me to look up on Google?"]

                speak(random.choice(G4))

                search\_query = takeCommand().strip()

                search\_google(search\_query)

            elif intent == "search\_youtube":

                Y1 = ["What should I search on YouTube?",

                      "Please tell me what to search on YouTube.",

                      "What do you want to search for on YouTube?",

                      "Can you specify what to search on YouTube?",

                      "What would you like me to look up on YouTube?",

                      "What video you want to see?"]

                speak(random.choice(Y1))

                search\_query = takeCommand().strip()

                search\_youtube(search\_query)

            elif intent == "send\_whatsapp\_message":

                speak("Please provide the recipient's name.")

                contact\_name = takeCommand()

                speak("What's the message?")

                message\_content = takeCommand()

                send\_whatsapp\_message(contact\_name, message\_content)

            elif intent == "make\_call":

                speak("Please provide the recipient's name.")

                contact\_name = takeCommand()

                make\_call(contact\_name)

            elif intent == "make\_video\_call":

                speak("Please provide the recipient's name.")

                contact\_name = takeCommand()

                make\_video\_call(contact\_name)

            elif intent == 'take\_photo':

                take\_photo()

            elif intent == 'take\_screenshot':

                take\_screenshot()

            elif intent == 'start\_screen\_recording':

                start\_screen\_recording()

            elif intent == 'stop\_screen\_recording':

                stop\_screen\_recording()

            elif intent == "thanks":

                speak("You're Welcome!")

            elif intent == "bye":

                speak("Bye sir, Have a good day.")

                break

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

    eel.start('page-1.html', size=(1500, 1500))

**• DESIGN:**

1. **Sign-Up Page:**

**A screenshot of a login form

Description automatically generated**

1. **Sign-In Page:**

**A screen shot of a computer

Description automatically generated**

1. **Main Interface:**

****

1. **FUTURE ENHANCEMENT**

The future of S.A.G.E. lies in its continuous evolution to meet the ever-changing needs of its users. Future developments will focus on incorporating advanced command tracking, which will allow the assistant to better understand and anticipate user needs based on previous interactions. Additionally, contextual understanding will be a key area of enhancement, enabling S.A.G.E. to interpret commands within the context of a conversation, thereby improving accuracy and relevance in its responses.

Personalized training will also be a significant focus, allowing S.A.G.E. to adapt to individual user preferences and usage patterns, offering a more tailored experience. Moreover, new features and integrations will be introduced to keep S.A.G.E. aligned with emerging technologies and user demands. This could include integrations with smart home devices, expanded multimedia capabilities, and more sophisticated information retrieval systems.

By continuously expanding its capabilities, S.A.G.E. will remain at the forefront of voice assistant technology, providing users with an intuitive and enhanced digital experience that evolves alongside technological advancements.

1. **SUMMARY & CONCLUSION**

**• Summary:**

S.A.G.E. (Smart Assistant for Greater Efficiency) is a sophisticated voice assistant designed to significantly boost productivity by automating a wide range of tasks through intuitive voice commands. The assistant offers a comprehensive suite of functionalities, including application control, file management, system operations, communication, web interaction, and information retrieval. Additionally, S.A.G.E. provides interactive features like games, reminders, alarms, and multimedia functions such as photo capturing, screen recording, and screenshots.

Utilizing TensorFlow for natural language processing and intent recognition, S.A.G.E. ensures accurate interpretation of commands and delivers precise responses. The assistant is capable of managing daily tasks with ease, offering real-time news and weather updates, suggesting entertainment options, and facilitating communication through platforms like WhatsApp. With thorough testing, detailed documentation, and a design that allows for future scalability, S.A.G.E. is a versatile and flexible tool suitable for both personal and professional use.

**• Conclusion:**

In conclusion, S.A.G.E. represents a dynamic and innovative approach to voice-assisted technology, designed to enhance productivity and streamline tasks through the use of advanced natural language processing powered by TensorFlow. By integrating a wide range of functionalities—from application control and file management to real-time updates and multimedia capabilities—S.A.G.E. offers a comprehensive solution for managing both personal and professional activities. Its intuitive design, coupled with thorough testing and scalability, ensures that S.A.G.E. is not only effective but also adaptable to the evolving needs of users in a rapidly changing digital landscape.

S.A.G.E. is more than just a voice assistant; it is a tool that redefines how users interact with their digital workspace, making technology more accessible, efficient, and user-friendly. Whether for managing routine tasks or handling complex operations, S.A.G.E. stands as a testament to the potential of automation in enhancing productivity and simplifying daily life.

**8. REFERENCES AND BIBLIOGRAPHY**

**• TensorFlow Documentation:**  
TensorFlow. (n.d.). *TensorFlow Documentation*. Retrieved from https://www.tensorflow.org/overview

**• Eel Documentation**:

Eel. (n.d.). *Eel Documentation*. Retrieved from [https://eeljs.org/](https://eeljs.org/" \t "_new)

**• Pycaw Documentation:**  
Pycaw. (n.d.). *Pycaw Documentation*. Retrieved from [https://github.com/AndreMikulec/pycaw](https://github.com/AndreMikulec/pycaw" \t "_new)

**• Pywinauto Documentation:**  
Pywinauto. (n.d.). *Pywinauto Documentation*. Retrieved from [https://pywinauto.readthedocs.io/](https://pywinauto.readthedocs.io/" \t "_new)

**OpenCV Documentation:**  
OpenCV. (n.d.). *OpenCV Documentation*. Retrieved from https://docs.opencv.org/