PERSONAL VOICE BASED ASSISTANT

FACULTY: B. RAJESH SIR

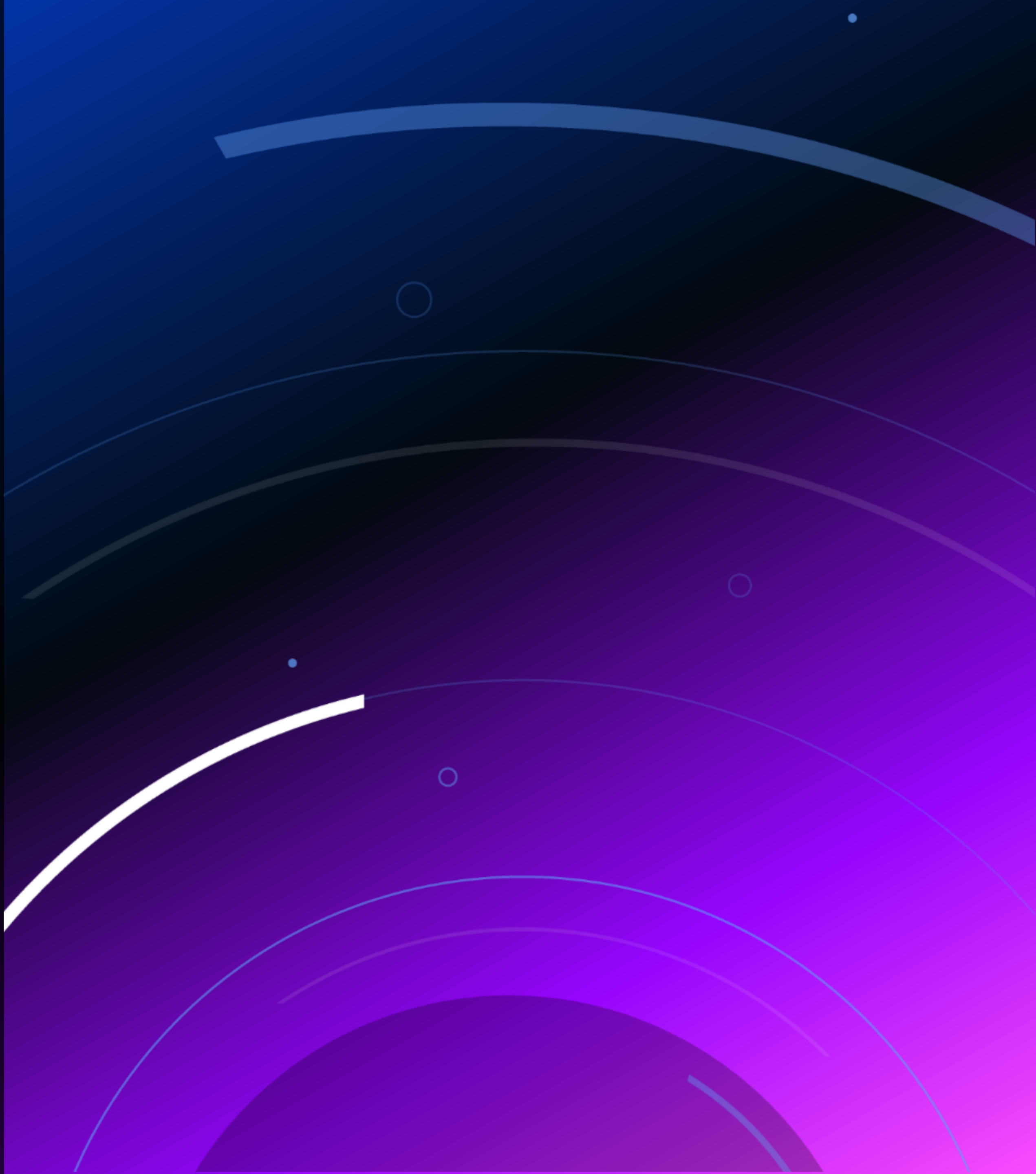
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GITAM UNIVERSITY, VISAKHAPATNAM

TEAM:

S.NO	NAME	REDG.NO
1	K.L.V.R.NAIDU	2024165348
2	K.BHAVYA	2024164904
3	N.AKASH RATAN	2024060023
4	ESHA PRANAVI	202408371

AGENDA



THE POWER OF CODING

INTRODUCTION:

JARVIS LITE IS A CLI-BASED SMART
VOICE ASSISTANT THAT INTERACTS WITH USERS
THROUGH NATURAL SPEECH, EXECUTING TASKS LIKE
OPENING APPLICATIONS, PERFORMING WEB SEARCHES,
TELLING TIME AND DATE, AND HANDLING CALCULATIONS
THROUGH VOICE COMMANDS.
IT USES **SPEECH RECOGNITION** FOR CAPTURING COMMANDS,
TEXT-TO-SPEECH (TTS) FOR RESPONSES, AND INTEGRATES
WITH VARIOUS SYSTEM-LEVEL FUNCTIONS LIKE LAUNCHING
APPS, USING THE CLIPBOARD, AND SIMULATING KEY PRESSES
TO ENHANCE USER PRODUCTIVITY IN A HANDS-FREE MANNER.

PROBLEM STATEMENT

- USERS OFTEN FIND IT DIFFICULT TO MULTITASK EFFICIENTLY.
- MANUAL EXECUTION OF ROUTINE TASKS IS TIME-CONSUMING AND BREAKS WORKFLOW CONTINUITY.
- NEED FOR A VOICE-BASED TOOL THAT HANDLES:

* APPLICATION LAUNCHING
* WEB SEARCHING
*TIME AND DATE QUERIES
* HANDS-FREE CALCULATIONS

JARVIS LITE SOLVES THIS PROBLEM. JUST USES VOICE COMMANDS AND PERFORMS TASKS INSTANTLY—NO TYPING, NO CLICKING, JUST SPEAK AND GET IT DONE!

FEATURES

MODULE	USE
os	To run system commands and open installed applications
webbrowser	To open URLs for Google searches or web-based apps
datetime	To fetch and format current time and date
speech_recognition	To capture and convert voice input into text
pyttsx3	For converting text to speech (offline Text-to-Speech engine)
pyperclip	To copy expressions to clipboard
time	To introduce delays between automated actions
pyautogui	To simulate keyboard shortcuts (e.g., Ctrl+V, Enter, Alt+F4)

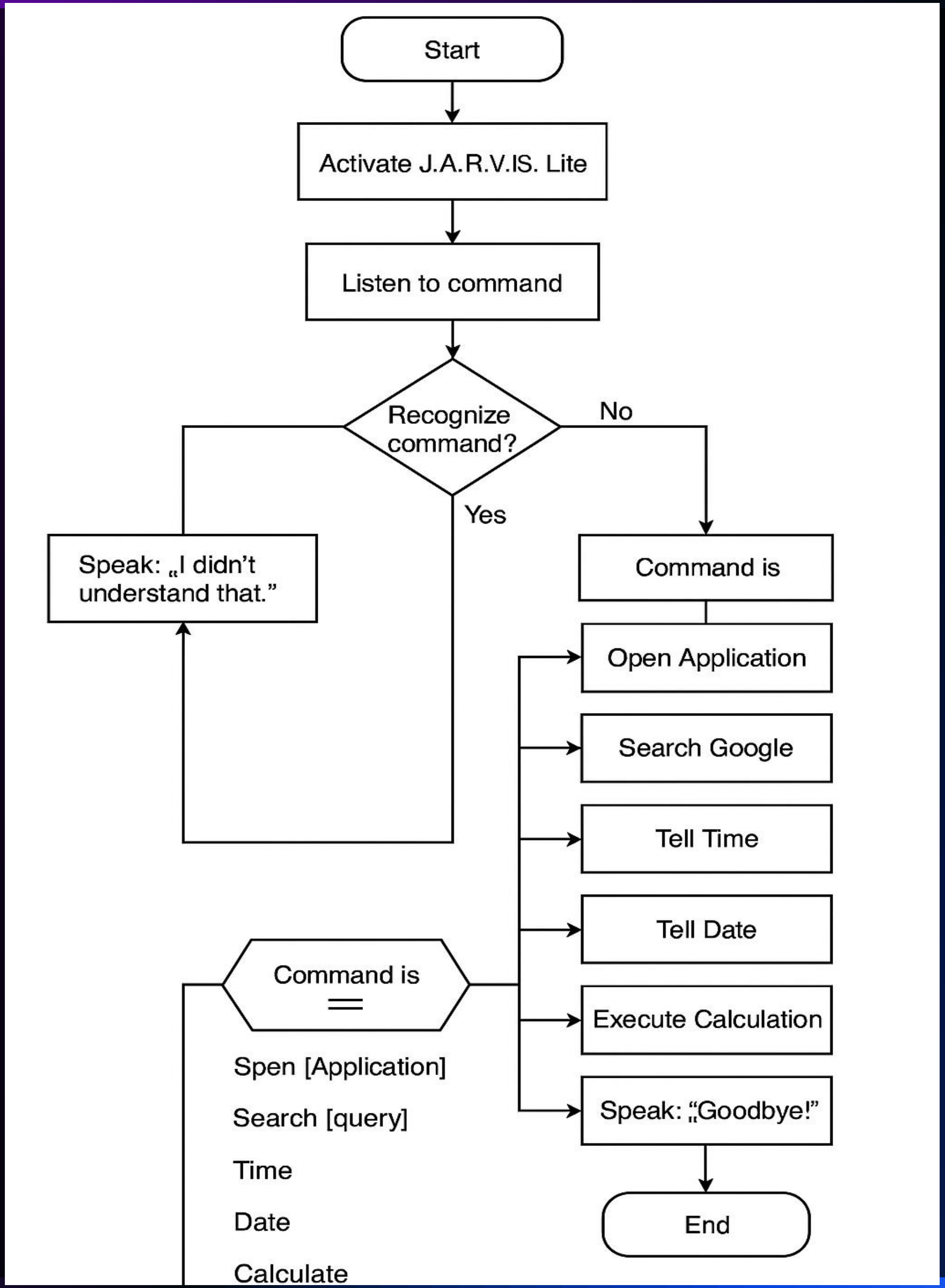
MODULES USED AND THEIR FUNCTIONS

Function	Purpose	Example Code
os.system()	Opens applications using system commands	os.system("notepad.exe")
webbrowser.open()	Opens web pages in default browser	webbrowser.open("https://www.google.com")
datetime.now()	Gets the current date and time	datetime.datetime.now()
strftime()	Formats date/time into readable strings	strftime("%H:%M:%S") for time, "%Y-%m-%d" for date
sr.Recognizer()	Initializes the speech recognizer	recognizer = sr.Recognizer()
recognizer.listen()	Captures audio from the microphone	audio = recognizer.listen(source)
recognizer.recognize_google()	Converts audio to text using Google Speech API	command = recognizer.recognize_google(audio)
pyttsx3.init()	Initializes the text-to-speech engine	engine = pyttsx3.init()
engine.say()	Queues text to be spoken	engine.say("Hello")
engine.runAndWait()	Speaks the queued text	engine.runAndWait()
pyperclip.copy()	Copies text to clipboard	pyperclip.copy(expression)
time.sleep()	Adds delay between actions	time.sleep(1)
pyautogui.hotkey()	Simulates key combinations (e.g., Alt+F4, Ctrl+V)	pyautogui.hotkey('ctrl', 'v')
pyautogui.press()	Simulates single key press	pyautogui.press('enter')

COMMANDS AND THEIR DESCRIPTION

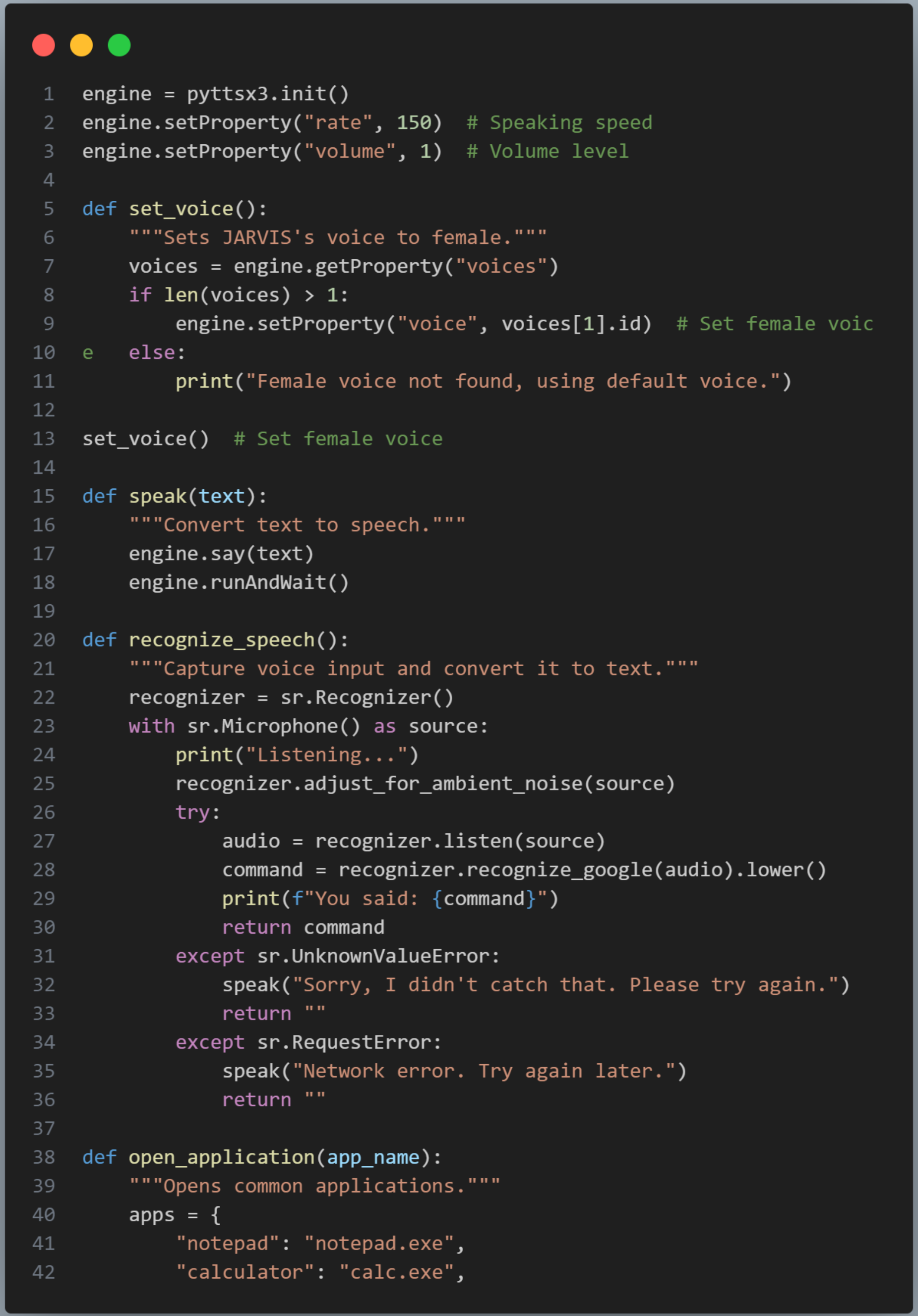
Command	Description
OPEN [Application]	Launches the specified application (e.g., Notepad, Chrome)
SEARCH [Query]	Performs a Google search for the given query
TIME	Tells the current time
DATE	Tells today's date
CALCULATE	Starts voice-based calculator for solving expressions
CLOSE CALCULATOR	Closes the calculator window
GOODBYE / OK BYE	Exits the JARVIS assistant with a farewell message

FLOWCHART


Start
Activate J.A.R.V.IS. Lite
Listen to command
Recognize command?
No
Yes
Speak: „I didn't understand that."
Command is
Open Application
Search Google
Tell Time
Tell Date
Execute Calculation
Speak: „Goodbye!"
End
Command is ==
Spen [Application]
Search [query]
Time
Date
Calculate

CODE SNIPPETS

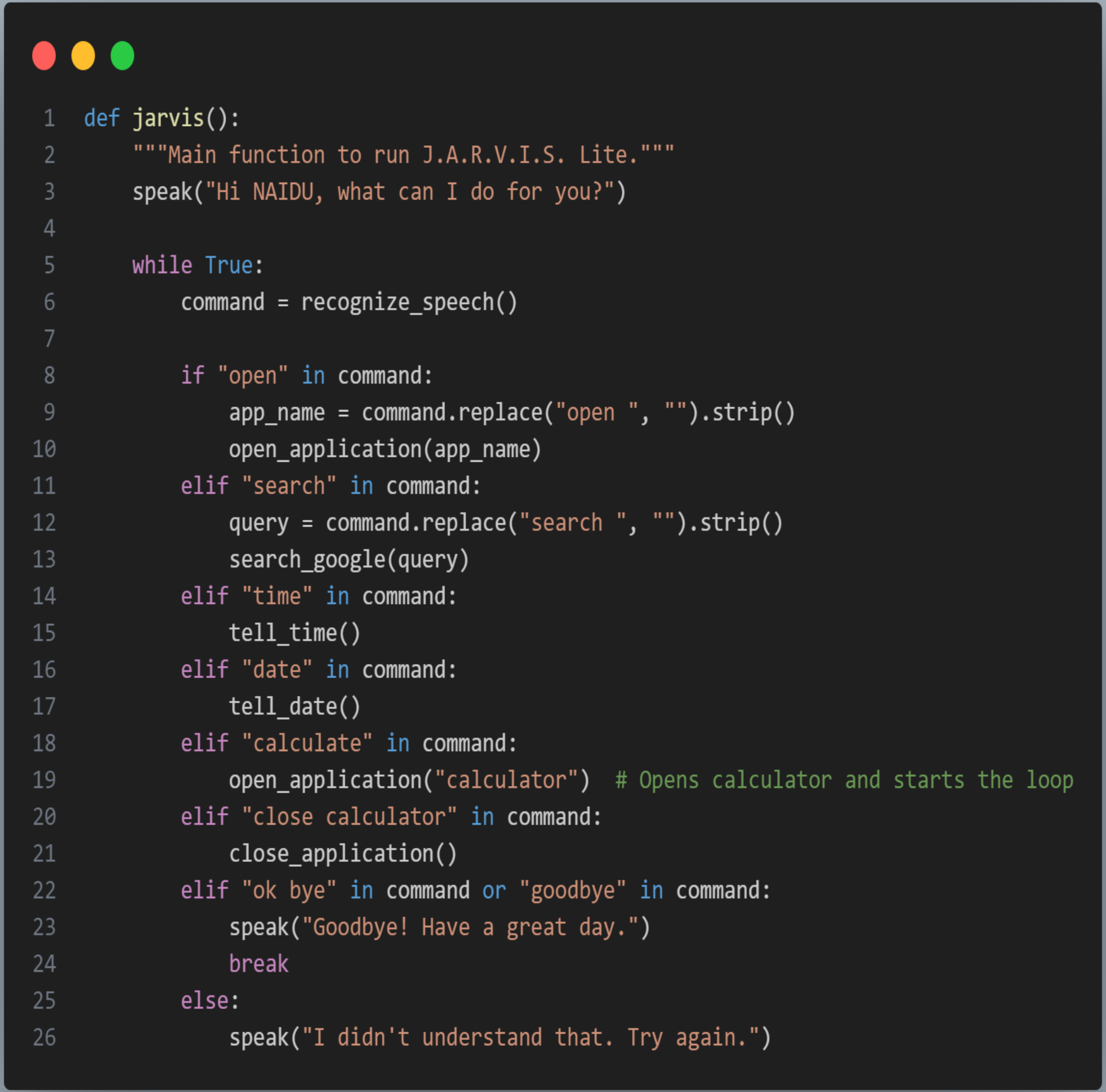
INITILISATION OF TEXT TO SPEECH ENGINE



```python
engine = pyttsx3.init()
engine.setProperty("rate", 150)  # Speaking speed
engine.setProperty("volume", 1)  # Volume level

def set_voice():
    """Sets JARVIS's voice to female."""
    voices = engine.getProperty("voices")
    if len(voices) > 1:
        engine.setProperty("voice", voices[1].id)  # Set female voic
e   else:
        print("Female voice not found, using default voice.")

set_voice()  # Set female voice

def speak(text):
    """Convert text to speech."""
    engine.say(text)
    engine.runAndWait()

def recognize_speech():
    """Capture voice input and convert it to text."""
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        recognizer.adjust_for_ambient_noise(source)
        try:
            audio = recognizer.listen(source)
            command = recognizer.recognize_google(audio).lower()
            print(f"You said: {command}")
            return command
        except sr.UnknownValueError:
            speak("Sorry, I didn't catch that. Please try again.")
            return ""
        except sr.RequestError:
            speak("Network error. Try again later.")
            return ""

def open_application(app_name):
    """Opens common applications."""
    apps = {
        "notepad": "notepad.exe",
        "calculator": "calc.exe",
```

MAIN LOOP

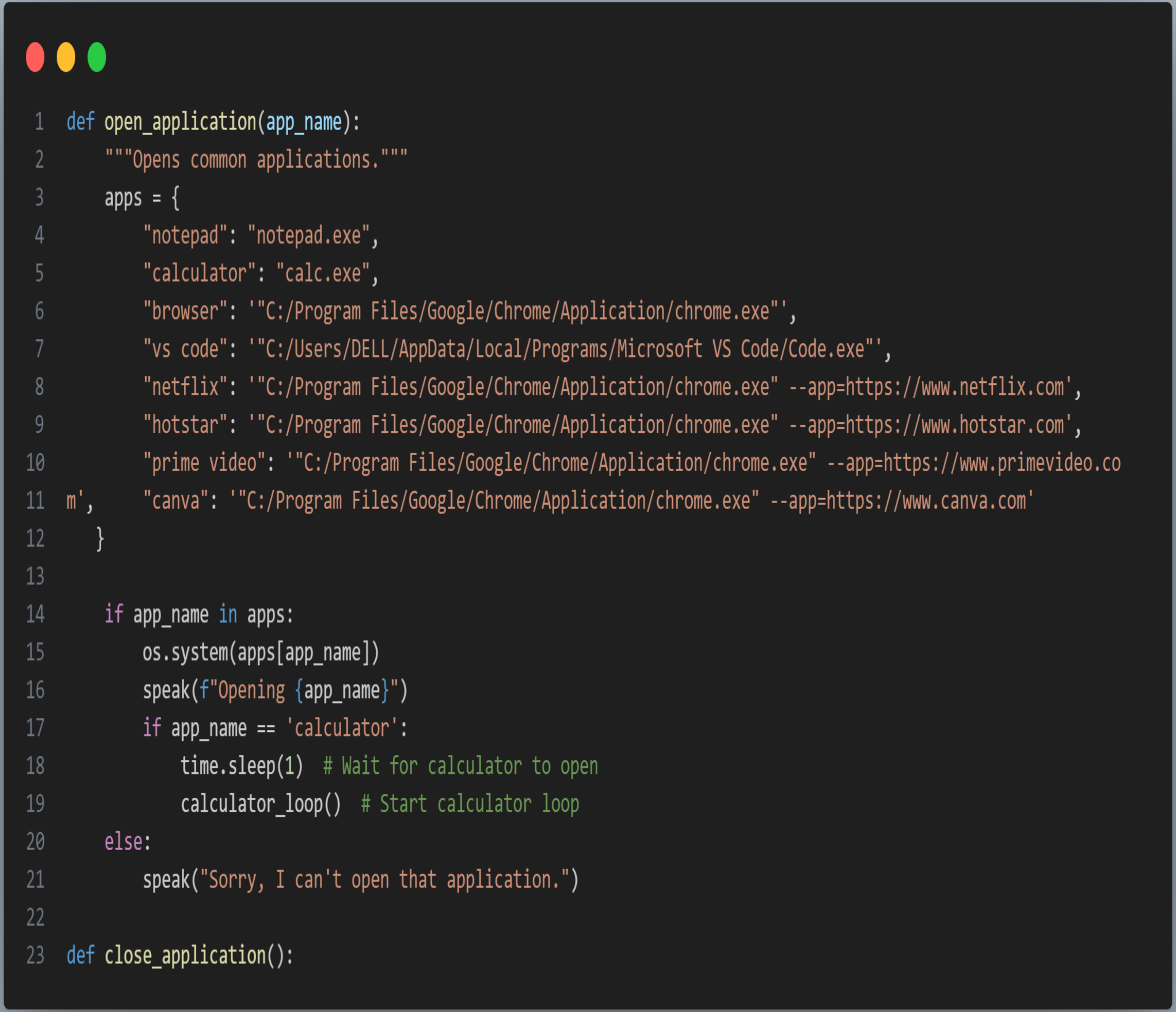
THE MAIN LOOP STARTS IMMEDIATELY AFTER WE RUN THE CODE AND ASKS OUR MESSAGE BASED ON OUR COMMAND IT GOES TO THE FUNCTION ,EVERY FUNCTION IS EXPLAINED BELOW



```python
def jarvis():
    """Main function to run J.A.R.V.I.S. Lite."""
    speak("Hi NAIDU, what can I do for you?")

    while True:
        command = recognize_speech()

        if "open" in command:
            app_name = command.replace("open ", "").strip()
            open_application(app_name)
        elif "search" in command:
            query = command.replace("search ", "").strip()
            search_google(query)
        elif "time" in command:
            tell_time()
        elif "date" in command:
            tell_date()
        elif "calculate" in command:
            open_application("calculator")  # Opens calculator and starts the loop
        elif "close calculator" in command:
            close_application()
        elif "ok bye" in command or "goodbye" in command:
            speak("Goodbye! Have a great day.")
            break
        else:
            speak("I didn't understand that. Try again.")
```

OPEN APPLICATION FUNCTION

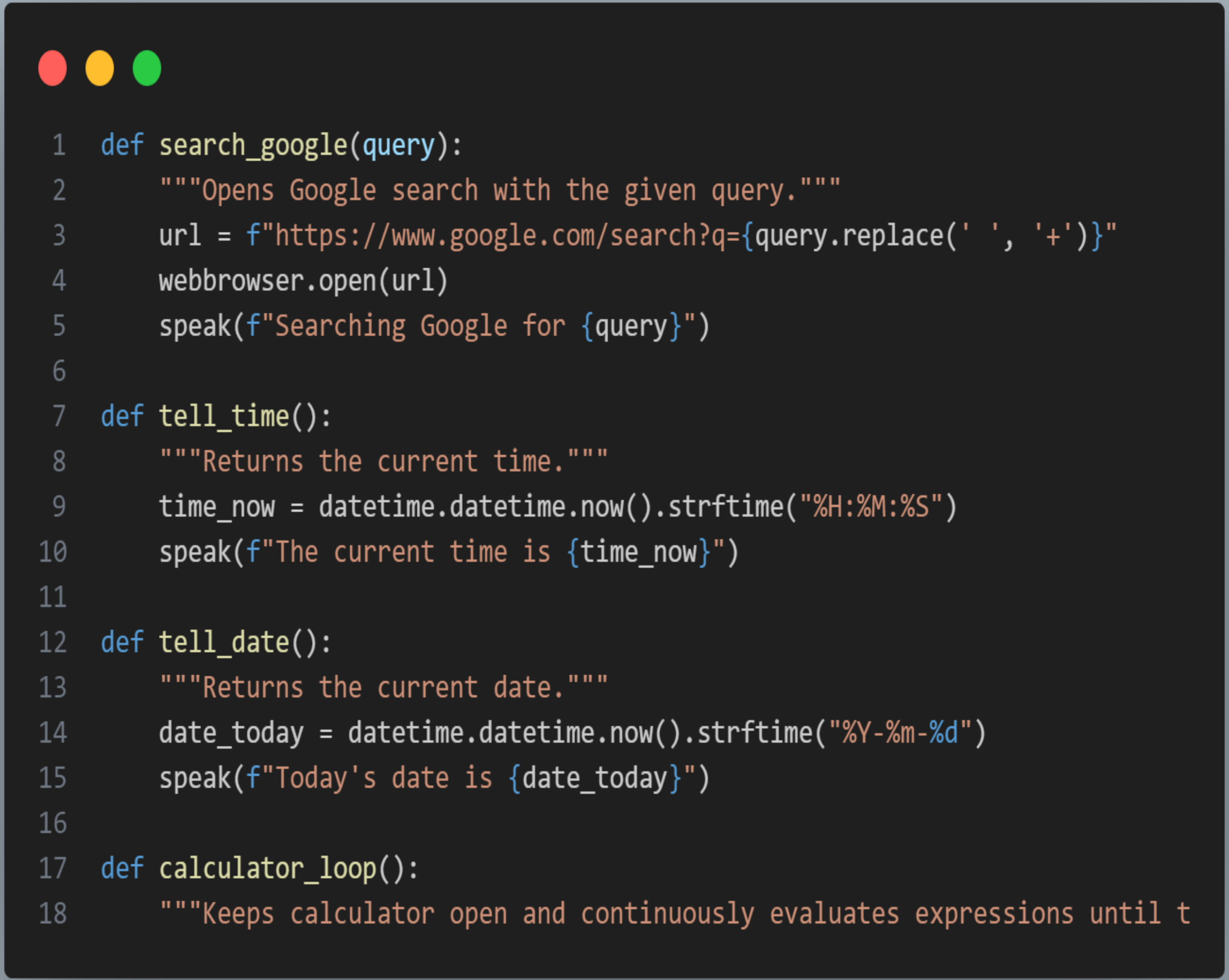
STORES FILE LOCATION AS VALUES WITH APP NAME AS ITS KEY
THEN WHEN SAID APP NAME
GOES TO VALUE TO PATH
AND OPENS THE APP
IF APP NOT THERE SENDS BACK A MESSAGE



```python
def open_application(app_name):
    """Opens common applications."""
    apps = {
        "notepad": "notepad.exe",
        "calculator": "calc.exe",
        "browser": '"C:/Program Files/Google/Chrome/Application/chrome.exe"',
        "vs code": '"C:/Users/DELL/AppData/Local/Programs/Microsoft VS Code/Code.exe"',
        "netflix": '"C:/Program Files/Google/Chrome/Application/chrome.exe" --app=https://www.netflix.com',
        "hotstar": '"C:/Program Files/Google/Chrome/Application/chrome.exe" --app=https://www.hotstar.com',
        "prime video": '"C:/Program Files/Google/Chrome/Application/chrome.exe" --app=https://www.primevideo.co
m',     "canva": '"C:/Program Files/Google/Chrome/Application/chrome.exe" --app=https://www.canva.com'
    }

    if app_name in apps:
        os.system(apps[app_name])
        speak(f"Opening {app_name}")
        if app_name == 'calculator':
            time.sleep(1)  # Wait for calculator to open
            calculator_loop()  # Start calculator loop
    else:
        speak("Sorry, I can't open that application.")

def close_application():
```

REMAINING USED FUNCTIONS

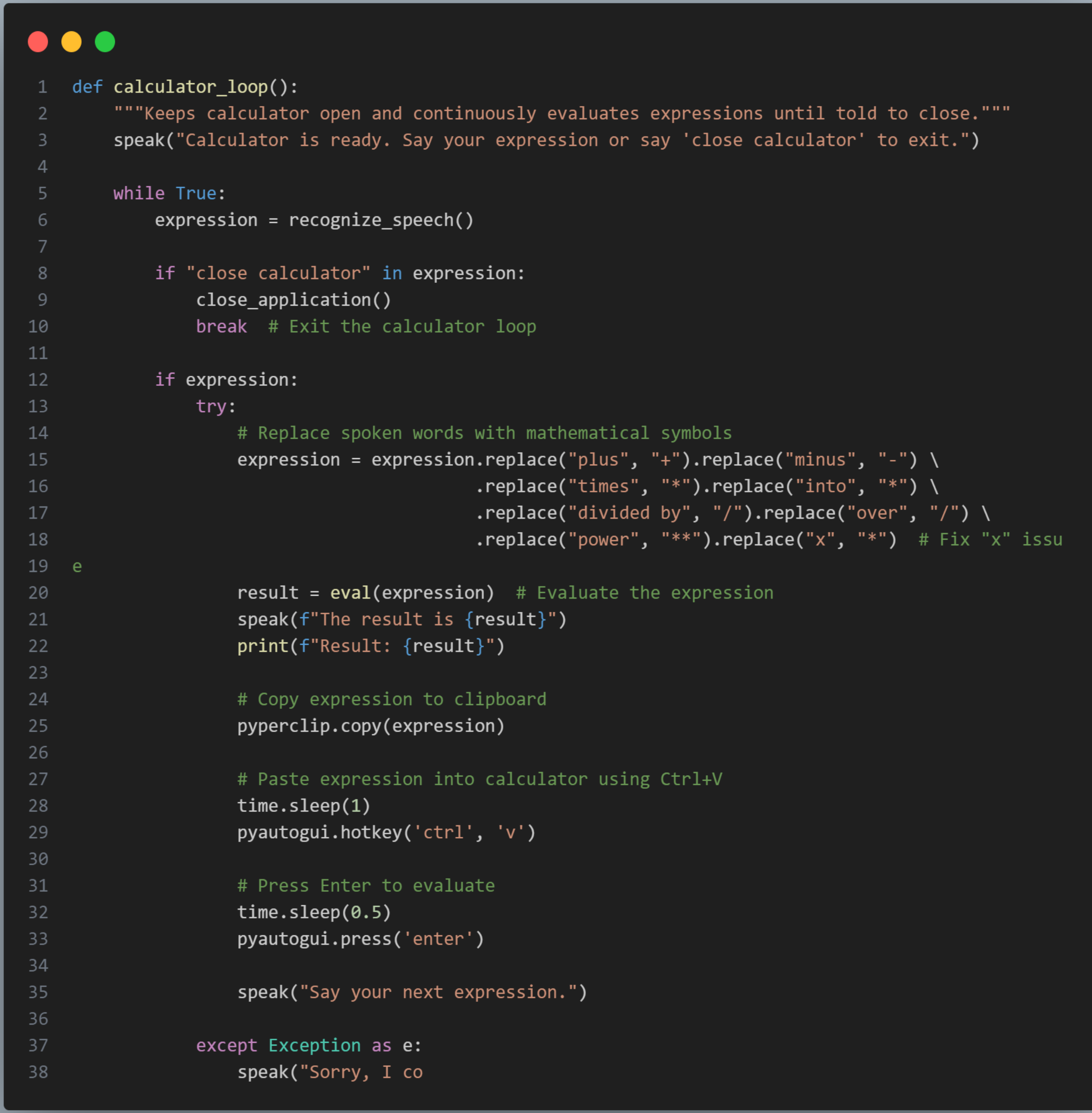
THESE ARE THE REMAINING FUNCTIONS LIKE SEARCH WHICH CONCATENATE QUERY WITH GOOGLE LINK AND AUTOMATICALLY OPENS THE PAGE,TELL TIME GIVE PRESENT TIME WITH MILLISECONDS AND TELL DATE GIVE THE TODAYS DATE



```python
def search_google(query):
    """Opens Google search with the given query."""
    url = f"https://www.google.com/search?q={query.replace(' ', '+')}"
    webbrowser.open(url)
    speak(f"Searching Google for {query}")

def tell_time():
    """Returns the current time."""
    time_now = datetime.datetime.now().strftime("%H:%M:%S")
    speak(f"The current time is {time_now}")

def tell_date():
    """Returns the current date."""
    date_today = datetime.datetime.now().strftime("%Y-%m-%d")
    speak(f"Today's date is {date_today}")

def calculator_loop():
    """Keeps calculator open and continuously evaluates expressions until t
```

CALCULATOR LOOP

THIS LOOP ASKS EXPRESSION OPENS CALCULATOR AUTOMATICALLY ENTERS EXPRESSION AND SHOWS RESULT AUTOMATICALLY BY PYPERCLIP AND PYAUTOGUI



```python
def calculator_loop():
    """Keeps calculator open and continuously evaluates expressions until told to close."""
    speak("Calculator is ready. Say your expression or say 'close calculator' to exit.")

    while True:
        expression = recognize_speech()

        if "close calculator" in expression:
            close_application()
            break  # Exit the calculator loop

        if expression:
            try:
                # Replace spoken words with mathematical symbols
                expression = expression.replace("plus", "+").replace("minus", "-") \
                                       .replace("times", "*").replace("into", "*") \
                                       .replace("divided by", "/").replace("over", "/") \
                                       .replace("power", "**").replace("x", "*")  # Fix "x" issu
e
                result = eval(expression)  # Evaluate the expression
                speak(f"The result is {result}")
                print(f"Result: {result}")

                # Copy expression to clipboard
                pyperclip.copy(expression)

                # Paste expression into calculator using Ctrl+V
                time.sleep(1)
                pyautogui.hotkey('ctrl', 'v')

                # Press Enter to evaluate
                time.sleep(0.5)
                pyautogui.press('enter')

                speak("Say your next expression.")

            except Exception as e:
                speak("Sorry, I co
```

ADVANTAGES:

*HANDS FREE INTERACTION
*EVEN DOES MULTITASKING
*OFFLINE TEXT TO SPEECH
BY PYTTSX3
*SMART APP LAUNCHER
*USER FRIENDLY INTERACTION
*BUILT IN AUTOMATED VOICE
CALCULATOR
*SIMPLE AND EDITABLE CODE
*INTERACTIVE EXIT FEATURE
(REPLIES A GOOD BYE MESSAGE)

THANK YOU