```python
#Copyright 2024 dking

#Used imports
import speech_recognition
import pyttsx3
import pyautogui
import time
import ctypes
import ctypes.wintypes
from AppOpener import open, close
from groq import Groq


#Constants, name of voice assistant and opening statement
name = 'dusty'
openingLine = "Hi sir"


#System User
user32 = ctypes.windll.user32


#CONSNTANT KEYS:
KEYEVENTF_EXTENDEDKEY = 0x0001
KEYEVENTF_KEYUP = 0x0002
VK_MEDIA_PLAY_PAUSE = 0xB3
VK_MEDIA_NEXT_TRACK = 0xB0
VK_MEDIA_PREV_TRACK = 0xB1
VK_VOLUME_UP = 0xAF
VK_VOLUME_DOWN = 0xAE
VK_VOLUME_MUTE = 0xAD


class Jarvis:

    def __init__(self):
        self.engine = pyttsx3.init()
        voices = self.engine.getProperty('voices')
        self.engine.setProperty('voice', voices[0].id)
        self.engine.runAndWait()
```

```
        self.recognizer = speech_recognition.Recognizer()
        self.groq_client = Groq(api_key= "")
        self.chat_history = [{
            "role": "system",
            "content": "You are a helpful assistant. You reply with very
short answers. For simple answers use 1-5 words, if I ask you to give a
complicated response use 2 sentences at most."
        }]

    def speak(self, text):
        pyttsx3.speak(text)

    def send_media_key(self, key_code):
        user32.keybd_event(key_code, 0, KEYEVENTF_EXTENDEDKEY, 0)
        time.sleep(0.1)
        user32.keybd_event(key_code, 0, KEYEVENTF_EXTENDEDKEY |
KEYEVENTF_KEYUP, 0)

    def handle_media_commands(self, text):
        if "play" in text or "pause" in text:
            self.send_media_key(VK_MEDIA_PLAY_PAUSE)
        elif "up" in text:
            self.send_media_key(VK_VOLUME_UP)
        elif "down" in text:
            self.send_media_key(VK_VOLUME_DOWN)
        elif "mute" in text:
            self.send_media_key(VK_VOLUME_MUTE)
        elif "skip" in text:
            self.send_media_key(VK_MEDIA_NEXT_TRACK)
        elif "rewind" in text:
            self.send_media_key(VK_MEDIA_PREV_TRACK)
        elif "previous song" in text or "last song" in text:
            self.send_media_key(VK_MEDIA_PREV_TRACK)
            self.send_media_key(VK_MEDIA_PREV_TRACK)
        elif "shuffle" in text:
            open("Spotify")
            time.sleep(.2)
            pyautogui.hotkey('ctrl', 's')
            pyautogui.hotkey('alt', 'tab')
        elif "like song" in text:
```

```
            open("Spotify")
            time.sleep(.2)
            pyautogui.hotkey('alt', 'shift', 'b')
            pyautogui.hotkey('alt', 'tab')

    def handle_app_commands(self, text):
        if "open" in text:
            app = text.replace("open", "").strip()
            open(app, match_closest=True)
            self.speak("Opening")
        elif "close" in text:
            app = text.replace("close", "").strip()
            close(app, match_closest=True)
            self.speak("Closing")

    def handle_llm_query(self, text):
        self.chat_history.append({"role": "user", "content": text})
        response = self.groq_client.chat.completions.create(
            model="llama3-70b-8192",
            messages=self.chat_history,
            max_tokens=8000,
            temperature=1.2
        )
        assistant_response = response.choices[0].message.content
        self.chat_history.append({
            "role": "assistant",
            "content": assistant_response
        })
        self.speak(assistant_response)
        print(name, assistant_response)

    def dailyTasks(self):
        open("Google Chrome")
        time.sleep(1)

pyautogui.write('https://docs.google.com/document/d/12dlCAvaaR0_V9p9FvMDjh
oEqajGm0-j7er7mvcQuSJo/edit')
        pyautogui.hotkey('enter')
        time.sleep(1)
        pyautogui.hotkey('ctrl', 't')
```

```
        pyautogui.write('Fun things happening in columbia today')
        pyautogui.hotkey('enter')
        time.sleep(1)
        pyautogui.hotkey('ctrl', 't')
        pyautogui.write('https://outlook.live.com/mail/0/')
        pyautogui.hotkey('enter')
        time.sleep(1)
        pyautogui.hotkey('ctrl', 't')
        pyautogui.write('https://web.groupme.com/chats')
        pyautogui.hotkey('enter')


    def listen(self):
        while True:
            try:
                with speech_recognition.Microphone() as mic:
                    self.recognizer.adjust_for_ambient_noise(mic,
duration=0.2)
                    audio = self.recognizer.listen(mic)
                    text = self.recognizer.recognize_google(audio).lower()
                    print(f"{text}")

                    if "nevermind" in text:
                        text = text.split("nevermind", 1)[1].strip()
                    elif "never mind" in text:
                        text = text.split("never mind", 1)[1].strip()

                    if name in text:
                        text = text.split(name, 1)[1].strip()

                        if "song" in text or "volume" in text or "mute" in
text:
                            self.handle_media_commands(text)
                        elif "open" in text or "close" in text:
                            self.handle_app_commands(text)
                        elif any(keyword in text for keyword in ["who",
"what", "where", "when", "why", "how"]):
                            self.handle_llm_query(text)
                        elif "daily task" in text:
                            self.dailyTasks()
```

```
                    elif "turn off" in text or "exit" in text:
                        return

            except speech_recognition.UnknownValueError:
                self.recognizer = speech_recognition.Recognizer()
                continue

    def run(self):
        self.speak(openingLine)
        self.listen()

if __name__ == "__main__":
    assistant = Jarvis()
    assistant.run()
```