import speech\_recognition as sr

import pyttsx3

import datetime

import webbrowser

import random

import requests

import json

import os

import time

from pygame import mixer

# Initialize the recognizer and text-to-speech engine

recognizer = sr.Recognizer()

engine = pyttsx3.init()

# Initialize pygame mixer for music playback

mixer.init()

# Simulated smart home devices

smart\_devices = {

"living room light": False,

"bedroom light": False,

"thermostat": 20,

"front door": "locked"

}

# Store reminders

reminders = []

def speak(text):

engine.say(text)

engine.runAndWait()

def listen():

with sr.Microphone() as source:

print("Listening...")

recognizer.adjust\_for\_ambient\_noise(source)

audio = recognizer.listen(source)

try:

print("Recognizing...")

query = recognizer.recognize\_google(audio, language='en-US')

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

return query.lower()

except sr.UnknownValueError:

print("Sorry, I didn't catch that. Could you please repeat?")

return ""

except sr.RequestError:

print("Sorry, I'm having trouble accessing the speech recognition service.")

return ""

def process\_command(command):

if "hello" in command:

speak("Hello! How can I help you today?")

elif "time" in command:

current\_time = datetime.datetime.now().strftime("%I:%M %p")

speak(f"The current time is {current\_time}")

elif "date" in command:

current\_date = datetime.datetime.now().strftime("%B %d, %Y")

speak(f"Today's date is {current\_date}")

elif "search" in command:

speak("What would you like me to search for?")

search\_query = listen()

if search\_query:

url = f"https://www.google.com/search?q={search\_query}"

webbrowser.open(url)

speak(f"Here are the search results for {search\_query}")

elif "joke" in command:

jokes = [

"Why don't scientists trust atoms? Because they make up everything!",

"Why did the scarecrow win an award? He was outstanding in his field!",

"Why don't eggs tell jokes? They'd crack each other up!"

]

speak(random.choice(jokes))

elif "weather" in command:

speak("Which city would you like to know the weather for?")

city = listen()

if city:

try:

api\_key = "YOUR\_API\_KEY\_HERE" # Replace with your OpenWeatherMap API key

url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api\_key}&units=metric"

response = requests.get(url)

data = json.loads(response.text)

temp = data['main']['temp']

desc = data['weather'][0]['description']

speak(f"The current temperature in {city} is {temp}°C with {desc}.")

except:

speak("I'm sorry, I couldn't fetch the weather information.")

elif "set reminder" in command:

speak("What would you like me to remind you about?")

reminder = listen()

if reminder:

reminders.append(reminder)

speak(f"Okay, I've set a reminder for: {reminder}")

elif "check reminders" in command:

if reminders:

speak("Here are your reminders:")

for i, reminder in enumerate(reminders, 1):

speak(f"{i}. {reminder}")

else:

speak("You have no reminders set.")

elif "play music" in command:

music\_dir = "path/to/your/music/directory" # Replace with your music directory path

songs = os.listdir(music\_dir)

if songs:

song = random.choice(songs)

speak(f"Playing {song}")

mixer.music.load(os.path.join(music\_dir, song))

mixer.music.play()

else:

speak("No music files found in the specified directory.")

elif "stop music" in command:

mixer.music.stop()

speak("Music stopped")

elif "smart home" in command:

if "light" in command:

room = "living room" if "living room" in command else "bedroom"

action = "on" if "on" in command else "off"

smart\_devices[f"{room} light"] = (action == "on")

speak(f"{room} light turned {action}")

elif "thermostat" in command:

if "increase" in command:

smart\_devices["thermostat"] += 1

elif "decrease" in command:

smart\_devices["thermostat"] -= 1

speak(f"Thermostat set to {smart\_devices['thermostat']} degrees")

elif "door" in command:

action = "unlocked" if "unlock" in command else "locked"

smart\_devices["front door"] = action

speak(f"Front door {action}")

else:

speak("Sorry, I didn't understand the smart home command.")

elif "exit" in command or "quit" in command:

speak("Goodbye!")

return False

else:

speak("I'm sorry, I don't understand that command. Can you please try again?")

return True

def main():

speak("Hello! I'm your voice assistant. How can I help you?")

while True:

command = listen()

if command:

if not process\_command(command):

break

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

main()