import pyttsx3

import time

import datetime

import speech\_recognition as sr

import webbrowser

# Function to use text-to-speech for communication

def speak(text):

engine = pyttsx3.init()

engine.say(text)

engine.runAndWait()

# Function to listen for user responses using speech recognition

def listen\_for\_response():

recognizer = sr.Recognizer()

with sr.Microphone() as source:

recognizer.adjust\_for\_ambient\_noise(source) # Adjust for ambient noise

print("Listening...")

audio = recognizer.listen(source)

try:

response = recognizer.recognize\_google(audio).strip().lower()

return response

except sr.UnknownValueError:

return "unknown"

except sr.RequestError:

return "unknown"

# Validate South African ID

def is\_valid\_sa\_id(id\_number):

return len(id\_number) == 13 and id\_number.isdigit()

# Prompt the user to enter their 13-digit South African ID manually

while True:

speak("Please enter your 13-digit South African ID manually, and press Enter when done:")

id\_number = input()

if is\_valid\_sa\_id(id\_number):

user\_id = id\_number

break

else:

speak("The ID entered should be a 13-digit South African ID. Please try again.")

# Define a list of daily routines

daily\_routines = [

"Take your morning medication.",

"Have a balanced meal for lunch.",

"Take a short walk in the afternoon.",

"Enjoy a nutritious dinner.",

"Remember to get a good night's sleep.",

]

# Create a dictionary to store user responses for each task

user\_responses = {task: "unknown" for task in daily\_routines}

# Greet the user and provide an introductory message

speak("Hello! I'm here to help you with your daily routines.")

speak("It's important to stay healthy and follow your daily routines.")

# Prompt the user to enter their name and surname

speak("Please enter your name and surname.")

name = listen\_for\_response()

# Validate and store the name and surname

if name != "unknown":

name\_parts = name.split()

if len(name\_parts) >= 2:

first\_name = name\_parts[0]

last\_name = name\_parts[1]

else:

first\_name = name\_parts[0]

last\_name = ""

else:

first\_name = ""

last\_name = ""

# Greet the user with their name and surname

if first\_name:

speak(f"Hello, {first\_name} {last\_name}! Welcome to the daily routines system.")

else:

speak("Hello! Welcome to the daily routines system.")

# Main loop for reminders and user interaction

for task in daily\_routines:

# Remind the user about the current task

speak(f"Now, it's time for you to {task}")

# Check if we have a previous response from the user for this task

previous\_response = user\_responses[task]

# Prompt the user with a yes/no question and listen for their response

speak("Have you completed this task? (yes/no)")

if previous\_response == "unknown":

while True:

response = listen\_for\_response()

if response == "yes" or response == "no":

user\_responses[task] = response

break

else:

speak("Please say 'yes' or 'no'.")

else:

response = previous\_response

# Provide feedback based on the user's response

if response == "yes":

speak("Great job! Keep it up.")

else:

speak("No worries, please complete the task when you can.")

speak("I'll check up on you after 30 minutes because it's my duty to make sure you stay healthy.")

time.sleep(1800) # Wait for 30 minutes (1800 seconds) before reminding again

# Google search functionality

speak("Is there anything else I can help you with? If you want to search on Google, just say 'search for' followed by your query.")

while True:

response = listen\_for\_response()

if response.startswith("search for"):

query = response[11:] # Extract the query after "search for"

speak(f"Searching Google for {query}...")

webbrowser.open(f"https://www.google.com/search?q={query}")

# Ask the user if they want to read the information found or not

speak("I found some information on the internet. Would you like me to read it to you?")

while True:

response = listen\_for\_response()

if "read" in response:

# Read the information found on the internet

speak("Sure! I'll read it to you.")

# You can add code here to fetch and read the information from the web

# Example: You can use web scraping libraries to extract and read the content

# For simplicity, let's use placeholder text to simulate reading

speak("Here's some information from the internet. Placeholder text for reading.")

# After reading, you can continue with user interactions

speak("Is there anything else I can assist you with?")

break

elif "continue" in response:

speak("Okay, you can continue reading. I'll be here if you need assistance.")

break

elif response == "unknown":

speak("Please say 'read' if you want me to read the information or 'continue' if you want to read it yourself.")

else:

speak("I'm not sure what you want. Please say 'read' or 'continue'.")

elif response == "unknown":

speak("Please say 'search for' followed by your query to perform a Google search.")

else:

speak("Is there anything else I can assist you with? If you want to search on Google, just say 'search for' followed by your query.")

# Final message

speak("You've completed all your tasks. If you need further assistance, feel free to ask.")