Alexa Voice Assistant - Python Source Code

```
'''■ Features This Version Supports:
Feature Description
■ Voice Input Use mic with wake-word "Alexa"
■ Music Search and open YouTube music
■ Open Websites open amazon.com, etc.
■ Google Search search <query>
■ Wikipedia wikipedia <topic>
■ Time & Date time, date
■ Jokes Random jokes
■ Logging Timestamped logs
■ TTS Speaks all responses
■ Dark GUI Clean, responsive layout
■ How to Use:
■ Click the microphone button to start listening
■■ Say "Alexa" followed by your command
■ Watch the visual feedback as it processes
■ See results in the activity log
■ Try commands by clicking them in the interface
■ Example Commands:
"Alexa, play Shape of You"
"Alexa, what's the weather in Mumbai?"
"Alexa, calculate 15 times 25"
"Alexa, open youtube.com"
"Alexa, tell me a joke"
"Alexa, what time is it?"
'''
import tkinter as tk
from tkinter import ttk, scrolledtext, messagebox
import pyttsx3
import speech_recognition as sr
from datetime import datetime
import webbrowser
import threading
import random
from youtubesearchpython import VideosSearch
class ModernVoiceAssistant:
def __init__(self, root):
self.root = root
self.root.title("■ Alexa Voice Assistant")
self.root.geometry("1200x800")
self.root.configure(bg="#1e1e2e")
self.root.minsize(1000, 700)
self.start_time = datetime.now()
self.is_listening = False
self.setup_styles()
self.setup_voice()
self.build_gui()
self.update_uptime()
def setup_styles(self):
style = ttk.Style(self.root)
style.theme_use('clam')
style.configure('TFrame', background='#1e1e2e')
style.configure('Title.TLabel', font=('Segoe UI', 24, 'bold'), foreground="#00d4ff", background="#1e1e2e")
style.configure('Sub.TLabel', font=('Segoe UI', 11), foreground="#cdd6f4", background="#1e1e2e")
style.configure('Log.TLabel', font=('Consolas', 10), foreground="#cdd6f4", background="#1e1e2e")
style.configure('TButton', font=('Segoe UI', 10), padding=6)
def setup_voice(self):
try:
self.recognizer = sr.Recognizer()
self.engine = pyttsx3.init()
self.engine.setProperty('rate', 150)
self.engine.setProperty('volume', 1)
voices = self.engine.getProperty('voices')
if voices:
self.engine.setProperty('voice', voices[1].id if len(voices) > 1 else voices[0].id)
except Exception as e:
messagebox.showerror("Voice Error", f"Failed to initialize voice engine: {e}")
def speak(self, text):
self.engine.say(text)
self.engine.runAndWait()
def build_gui(self):
```

```
# Top Frame
top_frame = ttk.Frame(self.root)
top_frame.pack(side=tk.TOP, fill=tk.X, padx=20, pady=10)
title = ttk.Label(top_frame, text="■ Alexa Voice Assistant", style='Title.TLabel')
title.pack(side=tk.LEFT)
self.uptime_label = ttk.Label(top_frame, text="Uptime: 00:00:00", style='Sub.TLabel')
self.uptime_label.pack(side=tk.RIGHT)
# Content Frame
content = ttk.Frame(self.root)
content.pack(fill=tk.BOTH, expand=True, padx=20, pady=10)
# Controls Panel
control_panel = ttk.LabelFrame(content, text="■ Controls", padding=10)
control_panel.pack(side=tk.LEFT, fill=tk.Y, padx=(0, 10))
self.mic_btn = tk.Button(control_panel, text="■ Start Listening", font=('Segoe UI', 14),
bg="#00d4ff", fg="white", relief="flat", command=self.toggle_listening)
self.mic_btn.pack(pady=20, fill=tk.X)
ttk.Button(control_panel, text="■ Play Music", command=lambda: self.run_command("play relaxing
music")).pack(fill=tk.X, pady=5)
ttk.Button(control_panel, text="■■ Weather", command=lambda: self.run_command("weather in
London")).pack(fill=tk.X, pady=5)
ttk.Button(control_panel, text="■ Time", command=lambda: self.run_command("time")).pack(fill=tk.X, pady=5)
ttk.Button(control_panel, text="■ Help", command=self.show_help).pack(fill=tk.X, pady=5)
# Log Panel
log_panel = ttk.LabelFrame(content, text="■ Activity Log", padding=10)
log_panel.pack(side=tk.RIGHT, fill=tk.BOTH, expand=True)
self.log_box = scrolledtext.ScrolledText(log_panel, wrap=tk.WORD, font=('Consolas', 10),
bg="#1e1e2e", fg="#cdd6f4", insertbackground='white')
self.log_box.pack(fill=tk.BOTH, expand=True)
self.log("system", "Assistant Initialized.")
def toggle_listening(self):
if not self.is_listening:
self.is_listening = True
self.mic_btn.config(text="■ Listening...", bg="#f38ba8")
threading.Thread(target=self.listen_for_command, daemon=True).start()
else:
self.is_listening = False
self.mic_btn.config(text="■ Start Listening", bg="#00d4ff")
def listen_for_command(self):
with sr.Microphone() as source:
self.log("system", "Listening...")
self.recognizer.adjust_for_ambient_noise(source)
try:
audio = self.recognizer.listen(source, timeout=5)
command = self.recognizer.recognize_google(audio).lower()
self.log("user", command)
if "alexa" in command:
command = command.replace("alexa", "").strip()
self.run_command(command)
else:
self.log("error", "Say 'Alexa' before your command.")
except sr.WaitTimeoutError:
self.log("error", "Listening timed out.")
except sr.UnknownValueError:
self.log("error", "Could not understand.")
except sr.RequestError as e:
self.log("error", f"Speech error: {e}")
finally:
self.toggle_listening()
def run_command(self, command):
threading.Thread(target=self.execute_command, args=(command,), daemon=True).start()
def execute_command(self, command):
response = self.process_command(command)
self.log("assistant", response)
self.speak(response)
def process_command(self, command):
command = command.lower()
if "play" in command:
song = command.replace("play", "").strip()
return self.play_song(song)
elif "weather" in command:
location = command.replace("weather in", "").strip()
webbrowser.open(f"https://www.google.com/search?q=weather+in+{location}")
return f"Showing weather for {location}"
elif "time" in command:
now = datetime.now().strftime("%I:%M %p")
```

```
return f"The time is {now}"
elif "date" in command:
return f"Today's date is {datetime.now().strftime('%B %d, %Y')}"
elif "joke" in command:
return random.choice([
"Why did the tomato turn red? Because it saw the salad dressing!",
"Why don't scientists trust atoms? Because they make up everything!",
"What do you call fake spaghetti? An impasta!"
])
elif "search" in command:
query = command.replace("search", "").strip()
webbrowser.open(f"https://www.google.com/search?q={query}")
return f"Searching Google for {query}"
elif "wikipedia" in command:
topic = command.replace("wikipedia", "").strip()
webbrowser.open(f"https://en.wikipedia.org/wiki/{topic}")
return f"Searching Wikipedia for {topic}"
elif "open" in command:
site = command.replace("open", "").strip()
url = f"https://{site}" if not site.startswith("http") else site
webbrowser.open(url)
return f"Opening {url}"
elif "exit" in command or "quit" in command:
self.root.quit()
return "Goodbye!"
return "Sorry, I didn't understand that."
def play_song(self, query):
try:
search = VideosSearch(query, limit=1)
result = search.result()
if result["result"]:
url = result["result"][0]["link"]
title = result["result"][0]["title"]
webbrowser.open(url)
return f"Playing: {title}"
else:
return "Couldn't find the song."
except Exception as e:
return f"Error finding song: {e}"
def log(self, who, message):
timestamp = datetime.now().strftime('%H:%M:%S')
self.log_box.insert(tk.END, f"[{timestamp}] {who.upper()}: {message}\n")
self.log_box.see(tk.END)
def update_uptime(self):
elapsed = datetime.now() - self.start_time
h, rem = divmod(int(elapsed.total_seconds()), 3600)
m, s = divmod(rem, 60)
self.uptime_label.config(text=f"Uptime: {h:02}:{m:02}:{s:02}")
self.root.after(1000, self.update_uptime)
def show_help(self):
help_text = (
"■■ Say 'Alexa' + command.\n\n"
"Examples:\n"
"- Alexa, play Shape of You\n"
"- Alexa, what time is it?\n"
"- Alexa, open google.com\n"
"- Alexa, joke\n"
"- Alexa, search Python tutorials\n"
)
messagebox.showinfo("Help", help_text)
if __name__ == "__main__":
root = tk.Tk()
app = ModernVoiceAssistant(root)
root.mainloop()
```