**INFOSYS SPRINGBOARD**

**TASK SUBMISSION**

**(ASSIGNMENT-2)**

**Submitted by:**

Shriya Vuppu

Batch 4

Under the Esteemed Guidance of

Mr . Bhargavesh Dakka

**TASK2: INTEGRATION OF LLM WITH VOICE AND SPEECH3**

OBJECTIVE:

The objective of this project is to integrate a Large Language Model (LLM) with voice-based input and output functionality. Using speech recognition to capture user input and text-to-speech to deliver responses creates an interactive voice assistant that can handle user queries effectively.

OUTCOMES:

1. Developed a voice-based assistant that listens to user input and responds using a predefined set of responses.

2. Integrated speech recognition, text-to-speech, and audio recording using the speech\_recognition, pyttsx3, and sounddevice libraries.

3. Learned how to use sounddevice to record audio instead of pyaudio.

4. Created a seamless interaction loop where the assistant listens, processes, and responds continuously until an exit command is given.

STEPS INVOLVED:

1. Creating the Project Folder and Virtual Environment

Project Folder: sales\_assistant\_project

Virtual Environment: Activated using sales\_assistant/activate/env

Virtual Environment: Created using python -m venv sales\_assistant/activate/env

2. Libraries to Install

1. speech\_recognition: For converting speech (audio) into text.
2. sounddevice: For recording audio (used as an alternative to pyaudio).
3. numpy: For handling audio data arrays.
4. pyttsx3: For converting text to speech.
5. requests: For making HTTP requests (if connecting to APIs).
6. json: To handle JSON data (typically included with Python's standard library).

Installation Commands

pip install SpeechRecognition

pip install sounddevice

pip install numpy

pip install pyttsx3

pip install requests

3.IMPLEMENTATION

import sounddevice as sd

import numpy as np

import speech\_recognition as sr

import pyttsx3

# Set the duration of the audio recording (in seconds)

duration = 5 # for a 5-second audio recording

# Set the sample rate (standard is 44100)

sample\_rate = 44100

# Initialize the text-to-speech engine

engine = pyttsx3.init()

# Predefined responses

def generate\_response(text):

responses = {

"hello": "Hello! How can I assist you?",

"how are you": "I'm fine, thank you! How about you?",

"what is your name": "I am your voice assistant.",

"what can you do": "I can help you with various tasks, like answering questions or recording notes.",

"exit": "Goodbye!"

}

# Normalize text to lowercase for matching

text = text.lower()

for key in responses:

if key in text:

return responses[key]

# Fallback response for unrecognized inputs

return "I'm not sure how to respond to that."

def record\_audio(duration, sample\_rate):

print("Recording...")

audio\_data = sd.rec(int(duration \* sample\_rate), samplerate=sample\_rate, channels=1, dtype='int16')

sd.wait() # Wait until the recording is finished

print("Recording complete.")

return audio\_data

# Function to recognize speech from the recorded audio

def recognize\_speech(audio\_data, sample\_rate):

recognizer = sr.Recognizer()

audio\_data = np.array(audio\_data, dtype=np.int16)

audio\_data = sr.AudioData(audio\_data.tobytes(), sample\_rate, 2)

try:

print("Recognizing speech...")

text = recognizer.recognize\_google(audio\_data)

print(f"Recognized Text: {text}")

return text

except sr.UnknownValueError:

print("Could not understand the audio.")

return None

except sr.RequestError:

print("Could not request results from the speech recognition service.")

return None

# Function to speak the recognized text

def speak\_text(text):

engine.say(text)

engine.runAndWait()

# Main loop for continuous interaction

while True:

# Record audio

audio\_data = record\_audio(duration, sample\_rate)

# Recognize speech

recognized\_text = recognize\_speech(audio\_data, sample\_rate)

if recognized\_text:

response = generate\_response(recognized\_text)

print(f"Assistant: {response}")

speak\_text(response)

# Exit the loop if the user says "exit"

if "exit" in recognized\_text.lower():

break

else:

print("No valid speech recognized.")

4.RUN THE SCRIPT

I have saved the file name as speech\_rec.py and executed in cmd by giving the command

Python speech\_rec.py

5.OUTPUT :

When you run the script, the assistant might interact as follows:

**1. Recording User's Speech**

The assistant will display a message that it is recording:

Recording...

Recording complete.

**2. Recognizing the Speech**

The assistant will recognize and convert your speech to text:

Recognizing speech...

Recognized Text: Hello

**3. Generating and Speaking the Response**

Based on the recognized text, the assistant will generate and speak a response:

Assistant: Hello! How can I assist you?

**4. Exiting the Program**

If you say "exit", the assistant will respond and stop the loop:

Assistant: Goodbye!

OUTPUT SCREENS



