Real Time Transcription

Overview

Introduction

The Vessel Real-time Transcription Websocket Service is a multi-cloud-compatible service designed to stream and transcribe audio data in real time. This service supports AWS, Azure, and GCP, enabling users to leverage their existing cloud infrastructure for transcription tasks. By utilizing WebSockets, it provides low-latency, high-throughput transcriptions suitable for various applications such as live captions, voice command processing, and more.

Ky Features

- **Multi-Cloud Support**: Users can choose between AWS, GCP, or Azure for real-time transcription.
- **Real-Time Transcription**: Instantly transcribes audio streams to text as they are received.
- **WebSocket Communication**: Enables low-latency and efficient bi-directional communication.
- **Multi-Language Support**: Transcribe audio in multiple languages and dialects.
- **Scalability**: Seamlessly scales with cloud infrastructure to handle varying loads.

Usage Scenarios

- **Live Captions**: Provide real-time subtitles for live events such as webinars, broadcasts, or conferences.
- **Voice Command Processing**: Capture and transcribe voice commands for smart devices and applications in real time.
- **Customer Support Transcription**: Transcribe customer support calls live for quality monitoring and record-keeping.
- **Meeting Transcriptions**: Automatically transcribe meeting audio in real-time for documentation and future reference.

Service Architecture

The Vessel Real-time Transcription Websocket Service is designed to be cloud-agnostic and operates using WebSocket protocols to ensure efficient and low-latency communication. It integrates with transcription services offered by AWS, GCP, and Azure.

Cloud Providers

- AWS (Amazon Web Services)
  - Utilizes AWS Transcribe for real-time transcription, providing robust and reliable service with support for a wide range of languages and dialects.
- GCP (Google Cloud Platform)
  - Employs Google Cloud Speech-to-Text API, offering advanced machine learning models to deliver high-quality real-time transcription capabilities.
- Azure (Microsoft Azure)
  - Leverages Azure Speech to Text service, delivering real-time transcription with support for numerous languages and high accuracy.

Service Usage

Prerequisites

1. Access to the Vessel Real time Transcription Service
2. Mulesoft API authentication credentials (Generate a Bearer token)

Connecting to the WebSocket

To use the Vessel Real-time Transcription Websocket Service, clients must establish a WebSocket connection to the service endpoint. The URL for the endpoint will vary depending on the chosen cloud provider.

Base URLs

The service is currently available only on AMER region.

- Dev: http://vsl-dev.pfizer.com/stt-realtime/
- Test: http://vsl-test.pfizer.com/stt-realtime/
- Prod: http://vsl.pfizer.com/stt-realtime/

Parameters

For the Vessel Real time transcription the parameters are being passes in the query string of the web socket url .

The following table describes the available parameters for the service .

Parameter name	Optional	Data type	Default value	Description
b_token	No	string	n/a	The authentication bearer token
vendor_id	Yes	string	azure	The cloud vendor which will be used for the transcription . The supported vendors are : azure , gcp , was .
src_lang	Yes	string	en-US	The language of the input audio stream .
audio_chunk_size	Yes	int	1	audio_chunk_size is for audio chunk size 1000 for 1 second. It should be integer
combined_chunks	Yes	int	1	combined_chunks define how many audio chunks you want to combine for transcription. It should be integer.
is_webportal	Yes	bool	False	is_webportal define output format if you want to output in form of combined text line by line then use True .other wise it will return only current audio chunk transcriprition. It should be True or False

Example Connection

Python packages

```
pip install asyncio
pip install pydub
```

```
import requests
import websockets
import asyncio
import os
from pydub import AudioSegment
from pydub.utils import make_chunks
import logging
import json
WS_BASE_URI = os.getenv("WS_BASE_URI","wss://vsl-dev.pfizer.com/stt-realtime/vws/v1/speech")
AUDIO_FILE ="4_sec.wav"
def get_access_token():
    url = "https://devfederate.pfizer.com/as/token.oauth2?grant_type=client_credentials"

    payload = f'client_id={os.getenv("STT_REAL_TIME_CLIENT_ID","xxxxxxxx")}&client_secret={os.getenv
("STT_REAL_TIME_CLIENT_SECRET","xxxxxxxx")}'
    headers = {
    'Content-Type': 'application/x-www-form-urlencoded',
    }

    response = requests.request("POST", url, headers=headers, data=payload)
    return response.json()["access_token"]
ACCESS_<a href="https://confluence.pfizer.com/plugins/servlet/pii4conf/pii?id=2034944"></a>()
VENDOR_ID = "gcp"
async def connect():

    url = f"{WS_BASE_URI}?vendor_id={VENDOR_ID}&b_token={ACCESS_TOKEN}"
    print(url)
    async with websockets.connect(url) as websocket:

        myaudio = AudioSegment.from_file(AUDIO_FILE, "wav")
        chunk_length_ms = 3000  # pydub calculates in millisec
        chunks = make_chunks(myaudio, chunk_length_ms)
        for i in chunks:
            i.export('temp.wav', format="wav")
            with open('temp.wav', 'rb') as fp:
                mydata = fp.read()

                await websocket.send(mydata)
                message = await websocket.recv()
                print(message)

asyncio.get_event_loop().run_until_complete(connect())
```

Sending Audio Data

Audio data should be sent as binary data (e.g., PCM). Ensure that the audio stream conforms to the specifications required by the selected cloud provider (sample rate, encoding, etc.).

The framerate of audio stream should be 16000Hz.

Receiving Transcriptions

Transcriptions are received as messages over the WebSocket connection. Each message contains transcribed text data that can be processed or displayed as needed.

Error Handling

Errors may occur during connection or transcription. It is important to implement robust error handling to manage these scenarios gracefully. Common error responses include:

- **Connection Errors**: Connection refused, timeout, or server unavailable.
- **Transcription Errors**: Unsupported audio format, speech recognition failure, or exceeding rate limits.

FAQS

Common questions and service limitations

Q: Which vendors are supported?

A: Below are the list of vendor and the type of service they offer

Vendor Name	Vendor ID
Google Cloud Platform	gcp
AWS(recommended)	aws

Q: What language pairs are currently supported?

A: Currently transcription is possible for the following input languages.

GCP	https://cloud.google.com/vision/docs/languages
AWS	https://docs.aws.amazon.com/elemental-live/latest/ug/captions-ocr-languages.html