# **LiveKit Voice Assistant Audio Length Validation**

## **1. Project Overview**

This document outlines the implementation of a LiveKit Voice Pipeline Agent with audio length validation capabilities. The system includes a mechanism that sends the length of generated audio to a dedicated server before Text-to-Speech (TTS) processing. If the audio length exceeds 60 seconds, the server returns the audio trimmed to its middle segment.

## **2. System Architecture**

The system consists of the following components:

**LiveKit Voice Pipeline Agent**: Handles voice interactions with users

**Audio Length Validation Server**: External server that validates and trims text

**Integration Layer**: Connects the Voice Pipeline Agent to the validation server

### **2.1 System Flow**

User speaks to the Voice Assistant

Speech is converted to text using STT (Speech-to-Text)

Text is processed by the LLM (Large Language Model)

Before TTS processing, text is sent to validation server

Validation server estimates audio length and trims if necessary

Trimmed text is returned to the Voice Pipeline Agent

Trimmed text is converted to speech using TTS

Audio is played back to the user

## **3. Implementation Details**

### **3.1 Voice Assistant Implementation**

The Voice Pipeline Agent is implemented using LiveKit's agent framework. The key components include:

**Voice Activity Detection (VAD)**: Silero VAD is used for detecting speech

**Speech-to-Text (STT)**: Deepgram's STT service with model selection based on participant type

**Language Model (LLM)**: OpenAI's language model for generating responses

**Text-to-Speech (TTS)**: OpenAI's TTS service for converting text to speech

**Audio Length Validation**: Custom implementation using an external server

### **3.2 Audio Length Validation**

The audio length validation is implemented as a callback function that is called before TTS processing. The before\_tts\_callback function:

Sends the text to be spoken to the validation server

Receives validated (potentially trimmed) text

Logs the original and validated text lengths

Returns the validated text for TTS processing

def before\_tts\_callback(assistant, text: str) -> str:  
 try:  
 logger.info(f"Original text sent for validation: {text[:50]}...")  
 response = requests.post(  
 '<https://19d2-183-82-34-206.ngrok-free.app/validate_audio_length>',  
 json={'text': text},  
 timeout=5  
 )  
   
 if response.status\_code == 200:  
 data = response.json()  
 validated\_text = data.get('validated\_text', text)  
 logger.info(f"Original length: {len(text)}, Validated length: {len(validated\_text)}")  
   
 if text != validated\_text:  
 logger.info("Text was trimmed by validation server")  
   
 return validated\_text  
 else:  
 logger.error(f"Validation server error: {response.status\_code}, {response.text}")  
 return text  
 except Exception as e:  
 logger.error(f"Error in before\_tts\_callback: {str(e)}")  
 return text

### **3.3 Integration with LiveKit**

The audio length validation is integrated into the LiveKit Voice Pipeline Agent using the before\_tts\_cb parameter:

agent = VoicePipelineAgent(  
 vad=ctx.proc.userdata["vad"],  
 stt=deepgram.STT(model=dg\_model),  
 llm=openai.LLM(),  
 tts=openai.TTS(),  
 chat\_ctx=initial\_ctx,  
 before\_tts\_cb=before\_tts\_callback,  
)

## **4. Validation Server**

The validation server is a separate service that receives text, estimates audio length, and trims if necessary.

### **4.1 API Endpoint**

**URL**: <https://19d2-183-82-34-206.ngrok-free.app/validate_audio_length>

**Method**: POST

**Request Body**: JSON with a text field

**Response**: JSON with a validated\_text field

### **4.2 Validation Logic**

The server implements the following logic:

Receives text from the Voice Pipeline Agent

Estimates audio length based on the text content

If estimated length > 60 seconds, trims the text to capture the middle segment

Returns the validated text

## **5. Deployment**

The system is deployed with the following components:

**LiveKit Voice Pipeline Agent**: Deployed as a Python application

**Validation Server**: Deployed with a public endpoint accessible via ngrok

## **6. Challenges and Solutions**

### **6.1 Callback Implementation**

**Challenge**: Initial implementation of the callback function wasn't properly integrated with the TTS pipeline.

**Solution**: Improved the callback function with proper error handling, logging, and return value management.

### **6.2 Text Trimming Logic**

**Challenge**: Ensuring the trimmed text maintains context and meaning.

**Solution**: Implemented middle-segment trimming to preserve the most important parts of the response.

### **6.3 Error Handling**

**Challenge**: Handling errors when the validation server is unavailable.

**Solution**: Added timeout to requests, comprehensive error handling, and fallback to original text when errors occur.

### **6.4 Integration Testing**

**Challenge**: Verifying that the trimmed text is correctly used in the TTS pipeline.

**Solution**: Added detailed logging of text lengths before and after validation to confirm trimming is applied correctly.

## **7. Future Improvements**

**Caching**: Implement caching for frequently used responses to reduce server load

**Advanced Trimming**: Enhance trimming logic to consider sentence boundaries and semantic importance

**Fallback Mechanism**: Implement a local fallback mechanism if the validation server is unavailable

**Monitoring**: Add monitoring for validation server performance and trimming statistics

## **8. Conclusion**

The implemented system successfully integrates audio length validation into the LiveKit Voice Pipeline Agent. The solution ensures that all spoken responses are kept within the 60-second limit by trimming text when necessary, while maintaining the context and meaning of the responses.