# **Developer Home Assignment: LiveKit Voice Pipeline with Audio Length Validation**

## Objective

The objective of this project is to develop a voice assistant using LiveKit's Voice Pipeline Agent. A key requirement is to validate the estimated length of the generated audio before it is processed by the Text-to-Speech (TTS) engine. If the length exceeds 60 seconds, the backend server trims the audio to its middle segment before returning it.

## Features Implemented

* **LiveKit Voice Pipeline Agent Integration**
* **Before-TTS Callback for Validation**
* **Flask Backend for Audio Length Validation**
* **Trimming of Long Audio Responses**
* **Summarization using Hugging Face Transformers (Bonus Feature)**
* **Exposing Flask Server with Ngrok for Accessibility**
* **UI for Voice Interaction and Demonstration**

## Implementation Details

### 1. LiveKit Voice Pipeline Setup

* Cloned and set up the LiveKit Voice Pipeline Agent demo from the official LiveKit GitHub repository.
* Installed all necessary dependencies and ensured the agent is fully operational.
* Integrated before\_tts\_cb callback to intercept text before being sent to the TTS engine.

### 2. Before-TTS Callback

* Extracts text from the input stream.
* Estimates the length of the audio based on the text input.
* Sends the estimated length to the Flask server for validation.
* If the response from the server contains modified text (e.g., trimmed or summarized), it replaces the original text before TTS processing.

### 3. Flask Backend Server

* Developed a RESTful API using Flask.
* API Endpoint: /validate\_audio to receive estimated audio length and text.
* If the estimated audio duration is over 60 seconds, it trims the middle portion of the text.
* Implements a summarization model using Hugging Face’s facebook/bart-large-cnn to further optimize lengthy responses.
* Returns the modified text to the LiveKit agent.

### 4. Communication Between LiveKit Agent and Flask Server

* Used requests library in Python to communicate between the LiveKit agent and the Flask server.
* ngrok is used to expose the local Flask server, enabling remote access during development and testing.

### 5. Voice UI for Interaction

* LiveKit's Voice Pipeline Agent is set up to handle voice interactions.
* Ensures a seamless user experience with clear and concise responses.

**Project Structure**

project-root/

│── livekit\_agent/

│ ├── minimal\_assistant.py # LiveKit Voice Pipeline Agent Implementation

│ ├── backend\_server\_flask.py # Flask Server for Audio Length Validation

│── docs/

│ ├── README.md # Project Documentation

│── requirements.txt # Dependencies

### API Details

**Endpoint: /validate\_audio**

* **Method:** POST
* **Request Body (JSON):**

{

"text": "Input text to be spoken",

"length": 75.0

}

* **Response (If length > 60 seconds):**

{

"message": "Trimmed and summarized text output"

}

* **Response (If length <= 60 seconds):**

{

"message": "Original text"

}