1. Arquitetura da Solução

Você precisará de três componentes principais interagindo:

1. **Seu Aplicativo (Cliente):** Responsável por capturar o microfone, codificar o áudio e enviar (Push) via RTMP para o IoT Hub. Também responsável por tocar o áudio que vem do dispositivo.
2. **Seu Backend (SaaS):** Responsável por autenticar o usuário e enviar os comandos de controle (API) para o IoT Hub.
3. **IoT Hub (Servidor):** Recebe o fluxo de áudio do App e o repassa para o Dispositivo (e vice-versa).

2. Desenvolvimento do Aplicativo (O substituto do OBS)

Seu aplicativo deve implementar uma biblioteca de **Streaming RTMP**.

- Para Android/iOS (Nativo ou Flutter/React Native):

Você deve utilizar bibliotecas como NodeMediaClient, LibRTMP ou FFmpeg mobile.

    - **Função:** Capturar o áudio do microfone do celular.
    - **Ação:** Conectar via RTMP na porta 1936 do IoT Hub e publicar o fluxo.

Configuração do Stream de Envio (App -> Servidor):

O App deve gerar uma URL dinâmica para cada sessão de conversa para evitar conflitos.

- **URL de Envio:** `rtmp://197.145.170.346:1936/live/{ID_DA_SESSAO}`
    - *Exemplo:* `rtmp://197.145.170.346:1936/live/chat_imei_868120`

3. Lógica de "Push-to-Talk" (O Fluxo de Código)

A implementação mais comum é o botão "Segurar para Falar" (Push-to-Talk). Veja o algoritmo lógico:

A. Quando o usuário pressiona o botão "Falar":

1. App: Começa a capturar o microfone e inicia o RTMP Publish para:

rtmp://137.131.170.156:1936/live/chat_{imei}.

2. **Backend:** Envia o comando para o dispositivo avisando onde buscar o áudio.

a. **Endpoint:** `/api/device/sendInstruct`
b. **Payload (`cmdContent`):**

JSON
```
{
    "startTalkURL": "rtmp://137.131.170.156:1936/live/chat_{imei}"
}
```

c. *Referência:* O comando `startTalkURL` notifica o dispositivo para puxar o stream do endereço especificado.

B. Enquanto o usuário está falando:

- O App continua enviando pacotes de áudio para o servidor.
- O Dispositivo conecta na URL recebida, faz o buffer e reproduz o áudio no alto-falante da cabine.

C. Quando o usuário solta o botão "Falar":

1. **App:** Para de enviar o stream (interrompe o RTMP Publish).
2. **Backend:** Envia o comando de parada para o dispositivo.
    a. **Payload (`cmdContent`):**

JSON
```
{
    "stopTalkURL": "rtmp://137.131.170.156:1936/live/chat_{imei}"
}
```

b. *Referência:* É necessário notificar o dispositivo para parar o streaming.

4. Como ouvir o Motorista (Monitoramento)

Simultaneamente ao processo acima, seu aplicativo deve estar "escutando" o dispositivo.

1. **Backend:** Envia o comando para o dispositivo iniciar a transmissão de vídeo/áudio ou apenas áudio (Monitor).
    a. Se usar a **API 37121**, defina `dataType: 0` (AV) ou 3 (Monitor).
2. **App:** Utiliza um Player compatível (Ex: **IJKPlayer**, **VLC Lib**, ou players nativos com suporte a FLV/RTMP) para conectar na saída do IoT Hub.

a. **URL de Leitura:** `http://137.131.170.156:8881/live/1/{IMEI}.flv`.

5. Requisitos Técnicos Críticos

Para que a implementação real funcione bem, atente-se a estes pontos encontrados na documentação:

1. **Codecs de Áudio:** O dispositivo JC450/JC400 espera formatos de áudio específicos. Ao configurar sua biblioteca de streaming no App, force o codec de áudio para **AAC** ou **PCM** com bitrate baixo (ex: 32kbps ou 64kbps) para garantir compatibilidade e baixa latência.
2. **Portas de Firewall:**
   a. Seu servidor deve aceitar conexões de entrada na porta **1936 (TCP)** vindas de *qualquer IP* (pois os celulares dos clientes estarão em redes móveis 4G/5G com IPs dinâmicos) .
   b. A porta **8881 (HTTP)** também deve estar pública para o App conseguir "baixar" o áudio do motorista .
3. **Latência:**
   a. Protocolos baseados em TCP (RTMP/HTTP-FLV) têm um delay natural de 2 a 5 segundos.
   b. Na implementação do Player no App, configure o **Buffer** para o mínimo possível (Zero ou Low Latency) para que a conversa flua quase em tempo real.

Resumo da Implementação

Ação	Tecnologia no App	Comando API (Backend)
Falar (App -> Carro)	Cliente RTMP (Push)	`startTalkURL`
Parar de Falar	Parar Cliente RTMP	`stopTalkURL`
Ouvir (Carro -> App)	Player FLV/RTMP (Pull)	`37121` (Start Live)