Project Report: Parallel Audio Stream Separation (Browser-Based)

Objective

To **capture** both system and microphone audio in a **Chrome browser** using Web APIs, and to **separate** them cleanly by removing system audio bleed/echo from the microphone stream — all in **real time**.

Goals Achieved

- Captured **system audio** via `getDisplayMedia({ audio: true })`
- Captured **microphone input** via `getUserMedia({ audio: true })`
- Processed both streams in real time using the **Web Audio API**
- Implemented a **basic LMS (Least Mean Squares) adaptive filter** to cancel echo from the mic stream
- Output two separate streams:
    - System audio (untouched)
    - Cleaned mic audio (echo-cancelled)

Root Cause of Echo/Bleed

- The **microphone picks up speaker output**, introducing system audio into the mic stream.
- This creates **overlap** (echo/bleed-in), especially during screen recording or remote collaboration.

Technical Stack

- **Language**: JavaScript (ES6+)
- **Browser APIs**:

`navigator.mediaDevices.getUserMedia` – mic input

`navigator.mediaDevices.getDisplayMedia` – system audio

Web Audio API:

`AudioContext`

`AudioWorklet` (for low-latency DSP)

Bundler: Vite (for local dev)

Dev Tools: Chrome, Console Logs

Project Structure

```
parallel-audio-separation/
├── src/
│   ├── index.html                # UI with Start button and audio elements
│   ├── main.js                   # Captures streams, sets up audio graph
│   ├── styles.css                # Minimal styling
│   └── audio/
│       ├── worklet-processor.js  # AudioWorkletProcessor for echo cancellation
│       ├── echoCanceller.js     # LMS DSP logic
│       └── utils.js             # (Optional) helpers for buffers, math
├── test/                         # (Optional) future unit tests
├── README.md                     # Setup and usage
└── vite.config.js                # Dev config for Vite
```

Key Implementation Details

System Audio Capture:Used `getDisplayMedia({ audio: true, video: true })` — but **removed the video track** immediately to avoid

screen capture:`stream.getVideoTracks().forEach(track => stream.removeTrack(track));`

Audio Routing:

`MediaStreamSource` nodes created for mic and system streams

Both routed to an `AudioWorkletNode` which runs the LMS filter

Echo Canceller:

LMS algorithm with:

Filter length: 128 taps

Step size (μ): 0.0005

Each audio frame is processed sample-by-sample in `process()` method

🔍 Issues & Fixes