🎧 Podcast Audio Separator — Technical Documentation

💡 Overview

Podcast Audio Separator is an internal tool designed to automatically separate two or more distinct voices from a podcast audio file. It utilizes advanced speech diarization to detect different speakers, then slices and exports their individual voice segments into separate audio files.

This tool streamlines post-production for podcast teams by eliminating manual audio separation, allowing editors to focus on mixing and content enhancement.

🧩 Architecture

The system follows a three-layer architecture that connects the frontend interface, backend API, and Python processing engine:

Workflow Diagram:
ReactJS (Frontend) → API Request → NodeJS Express (Backend) → Python Pyannote (Speech Diarization)

1. **ReactJS (Frontend)**
   - Handles the file upload and user interface for progress tracking.
   - Sends API requests to the backend for processing.
2. **NodeJS Express (Backend)**
   - Receives uploaded audio files and temporarily stores them.
   - Invokes the Python script that performs diarization.
   - Handles slicing of audio based on timestamps and returns results.
3. **Python (Pyannote Library)**
   - Performs **speaker diarization**, identifying segments for each speaker.
   - Returns timestamps and speaker labels to the backend for slicing.

🗃️ Database Schema

No database interaction is required.
All operations are performed in-memory or through temporary local storage on the server.

⚙️ Setup

Requirements

- NodeJS (v18 or later)
- Python (v3.10 or later)
- ReactJS (for building frontend)
- PM2 (for process management on server)
- FFmpeg (for audio format conversions)

Installation Steps

Clone the repository

```
git clone https://your-repo-url.git
cd podcast-audio-separator
```

1. **Setup the Backend**

   ```
   cd backend
   npm install
   pm2 start server.js --name "podcast-backend"
   ```

2. **Setup the Frontend**

   ```
   cd frontend
   npm install
   npm run build
   ```

3. **Setup the Python Environment**

   ```
   cd python
   pip install -r requirements.txt
   ```

4. **Run the system**
   - Ensure all services are running (Frontend build, NodeJS backend, Python scripts).

   - Access via your server’s public IP or domain.

📖 User Manual

1. **Upload File**
   - Select your podcast audio file (MP3/WAV).
   - Wait until the app finishes extracting and visualizing the audio waveform.

2. **Click “Diarize”**
   - The system runs speech diarization to detect different speakers.
   - Wait until processing is complete.

3. **Click “Separate Voices”**
   - The tool slices the audio into individual voice files based on diarization data.

4. **Click “Download Audio Files”**
   - Once finished, download the separated voice tracks as ZIP or individual files.