Overview

This project processes an audio file containing multiple speakers, identifies different speakers, and generates a speaker-labeled text transcript. The system processes the audio in a single step using `complete_file.py`, applying speaker diarization and transcription.

Project Workflow

The project has been simplified to a single step:

- Run `complete_file.py` with an audio file as input:

```
python complete_file.py <audio_file>
```

- The script processes the audio by:
  1. Converting it to a standardized format.
  2. Applying speaker diarization to identify speakers.
  3. Using Whisper to transcribe speech.
  4. Generating a labeled transcript with speaker annotations.
- The final transcript is saved as `<audio_file_name>_transcript.txt`.

Technologies Used

- **Python** for scripting and automation.
- **Pydub** for audio processing.
- **Pyannote Audio** for speaker diarization.
- **Whisper (OpenAI)** for transcribing speech to text.
- **FFmpeg** for handling audio file formats.
- **dotenv** for loading environment variables securely.

Output and Results

- `<audio_file_name>_transcript.txt` – The final labeled transcript containing speaker-wise text.

Setting Up Environment Variables

The script requires a Hugging Face authentication token stored in a `.env` file.

Steps to Obtain and Set Up the Token:

1. **Get the Token**:
   - Go to Hugging Face
   - Sign in and navigate to your profile settings.
   - Generate an API token under the 'Access Tokens' section.
2. **Create a `.env` File** in the Project Directory:

```
echo "HF_TOKEN=your_huggingface_token_here" > .env
```

3. The script automatically loads the token from `.env`.

Use Cases

- Transcribing and labeling speakers in meetings and interviews.
- Generating subtitles for podcasts, panel discussions, and talk shows.
- Creating AI-powered meeting notes.
- Enhancing accessibility for recorded conversations.