Documentation

1. Prerequisites

Before starting, ensure you have the following:

Checking Python Version and Installation

To check if Python is installed, run:

```
python --version
```

If Python is not installed or a version other than 3.11.x is installed, uninstall the existing version and install Python 3.11.9.

Windows:

1. Uninstall previous Python versions:
   - Open Control Panel > Programs and Features.
   - Find Python, right-click, and select Uninstall.
2. Download and install Python 3.11.9 from Python's official website.
3. Ensure Python is added to the system PATH during installation.

Linux:

1. Uninstall previous Python versions:

```
sudo apt remove python3 && sudo apt autoremove
```

2. Install Python 3.11.9:

```
sudo apt update && sudo apt install python3.11
```

3. Verify installation:

```
python --version
```

2. Setting Up the Virtual Environment

A virtual environment helps keep dependencies isolated.

Windows:

```
cd path\to\your\project-folder
python -m venv venv
venv\Scripts\activate
```

Linux:

```
cd /path/to/your/project-folder
python -m venv venv
source venv/bin/activate
```

3. Installing Required Dependencies

Run the following command inside the virtual environment:

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

If `requirements.txt` is missing, install packages manually:

```
pip install pydub pyannote.audio openai-whisper librosa matplotlib ffmpeg
python-dotenv
```

Installing FFmpeg

FFmpeg is required for audio processing.

Windows:

1. Download FFmpeg from ffmpeg.org
2. Add `ffmpeg` to the system PATH.

Linux:

```
sudo apt update && sudo apt install ffmpeg
ffmpeg -version
```

4. Project Workflow - Running the Project

Now that everything is set up, run the project using:

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

Expected Output

The script will process the given audio file and generate a transcript as:

```
<audio_file_name>_transcript.txt
```

5. Understanding the Code

The project consists of a script: `complete_file.py`.

How It Works:

1. Loads environment variables: Reads `HF_TOKEN` from a `.env` file.
2. Validates input file: Ensures the audio file exists.
3. Converts audio: Transforms the file into a 16kHz mono WAV format.
4. Runs speaker diarization: Identifies speakers using `pyannote.audio`.
5. Transcribes speech: Uses Whisper to transcribe each speaker's segment.
6. Saves results: Outputs a transcript to `<audio_file_name>_transcript.txt`.
7. Cleans up temporary files: Deletes temporary WAV files.

6. Setting Up the .env File

The Hugging Face API token should not be hardcoded. Instead, create a `.env` file.

```
touch .env
```

Open `.env` and add your Hugging Face token:

```
HF_TOKEN=your_huggingface_token_here
```

Obtaining a Hugging Face Token

1. Go to Hugging Face.
2. Log in or create an account.
3. Navigate to **Settings > Access Tokens**.
4. Generate a new token and copy it.

5. Paste the token in the `.env` file.

After setting up the `.env` file, the script will automatically load it.

7. Streamlit Dashboard

We have also created a **Streamlit dashboard** to provide an interactive interface for uploading and processing audio files.

Running Streamlit Locally (For Testing)

To test the dashboard locally, activate your virtual environment and run:

```
streamlit run app.py
```

This will launch a local web app where you can upload `.wav` files, process them, and download the generated transcript.

`app.py` Overview

This script provides a web-based interface to upload, process, and clear uploaded files.

8. Hosting the Project on Streamlit Cloud

To deploy the project on **Streamlit Cloud**, follow these steps:

1. **Push the project to GitHub**

```
git init
git add .
git commit -m "Initial commit"
git branch -M main
git remote add origin https://github.com/yourusername/your-repo.git
git push -u origin main
```

2. **Go to** Streamlit Cloud and sign in.
3. Click **New app** and select your repository.
4. Set `app.py` as the entry point.
5. Click **Deploy**.

Now your application will be available online, allowing users to upload and process audio files via a web interface.

This documentation now includes **development, local testing, and final hosting** instructions. 🚀