Voice Authentication System Documentation (Resemblyzer)

1 Overview

This documentation outlines the setup, installation, and usage of a Python-based voice authentication system utilizing the **Resemblyzer** library. This approach creates a lightweight voice embedding system that works efficiently on Python 3.11+ without the heavy dependencies of SpeechBrain.

2 System Architecture

The system operates by converting voice audio into mathematical vectors (embeddings) using a deep neural network.

- **Registration:** The user records voice samples. The system processes these samples to generate a 256-dimensional vector representing the unique characteristics of the user's voice.
- **Storage:** Vectors are stored locally in a JSON file (`audio_db/users.json`).
- **Verification:** A new audio sample is recorded and converted into a vector. The system calculates the **Cosine Similarity** between the new vector and the stored vectors. If the similarity score exceeds a defined threshold (default 0.75), access is granted.

3 Dependencies

This project requires the following Python libraries:

3.1 Core Libraries

Library	Role and Technical Detail
Resemblyzer	The core engine. Provides a pre-trained deep learning model (ECAPA-TDNN and SincNet) for speaker verification. Processes raw audio (16kHz) and outputs a 256-dimensional embedding.
NumPy	Fundamental for scientific computing. Handles high-dimensional arrays and cosine similarity calculations.
SoundDevice	Provides real-time audio I/O. Wraps PortAudio for streaming live audio data.

3.2 Audio Handling & Utilities

Library	Role and Technical Detail
SoundFile	Audio file reading and writing. Wraps libsndfile.
Colorama	Terminal output styling. Ensures cross-platform ANSI color compatibility.
Pathlib	Object-oriented filesystem path manipulation.

4 Installation Guide

4.1 Prerequisites

- Operating System: Windows, macOS, or Linux.
- Python Version: 3.11 or 3.12 (Recommended).
- Microphone: A working microphone.
- Audio Drivers: PortAudio drivers (usually included with OS).

4.2 Environment Setup (Optional but Recommended)

It is best practice to use a virtual environment.

4.2.1 Windows (PowerShell)

```
cd E:\voice-auth
python -m venv venv
.\venv\Scripts\Activate.ps1
```

4.2.2 macOS/Linux

```
cd voice-auth
python3 -m venv venv
source venv/bin/activate
```

4.3 Install Dependencies

```
pip install --upgrade pip
pip install resemblyzer sounddevice soundfile numpy colorama
```

5 Running the Application

1. Create a file named `voice_auth_simple.py` and paste the provided Python code.
2. Run the script:

```
python voice_auth_simple.py
```

6 User Guide

6.1 Main Menu

- Register New User
- Authenticate User
- List All Users
- Exit

6.2 Registration Process

1. Select **Option 1**.
2. Enter a unique **username**.
3. Record **2 voice samples** (default).
4. Follow prompts to record and save embeddings.

6.3 Authentication Process

1. Select **Option 2**.
2. Enter the **username** to verify.
3. Record a voice sample.
4. The system calculates the similarity score.

7 Troubleshooting

Error	Solution
ModuleNotFoundError: No module named 'sounddevice'	Install via pip. On Linux, install system libraries `install libportaudio2`.
Error: Default input device not found	Check OS sound settings and microphone connec
Low Authentication Scores	Ensure a quiet environment. Record more samp tration.
PyTorch Issues	Use the standard CPU version of PyTorch.

8 File Structure

After running the application, your project directory will look like this:

```
E:\voice-auth\
|-- voice_auth_simple.py
|-- venv/
|-- audio_db/
    |-- users.json
    |-- user1/
        |-- sample_1.wav
        |-- sample_2.wav
    |-- temp/
        |-- verify_temp.wav
```