Transcriber - Voice Recognition Project

A Python-based voice recognition system that provides both online and offline transcription capabilities with automatic fallback based on internet connectivity.

Project Overview

This project implements a smart voice transcription system that automatically detects internet connectivity and chooses the appropriate transcription method:

- **Online Mode**: Uses Google Speech Recognition API for high-accuracy transcription
- **Offline Mode**: Designed for local transcription (currently in development)

Features

- **Automatic Connectivity Detection**: Checks internet connection and selects appropriate transcription method
- **Real-time Voice Recognition**: Continuous listening and transcription until user says "exit"
- **Ambient Noise Adjustment**: Automatically adjusts for background noise
- **Error Handling**: Comprehensive error handling for various speech recognition scenarios
- **Modular Design**: Separate modules for online and offline transcription

Project Structure

```
Transcriber/
├── VoiceRecognition/
│   ├── main.py                # Main application entry point
│   ├── online_googleAudio.py   # Google Speech Recognition
implementation
│   └── offline_whisper.py      # Offline transcription (placeholder)
├── .gitignore                 # Git ignore rules for Python projects
└── README.md                  # This file
```

File Descriptions

`main.py`

The main entry point that:

- Imports required modules and functions
- Checks internet connectivity using Google's DNS server (8.8.8.8)
- Initializes the speech recognizer
- Routes to appropriate transcription method based on connectivity

`online_googleAudio.py`

Implements online transcription using Google Speech Recognition:

- Continuous listening loop with microphone input
- Ambient noise adjustment for better accuracy
- Real-time transcription display
- Exit command recognition ("exit" to stop)
- Comprehensive error handling for network and recognition issues

`offline_whisper.py`

Placeholder for offline transcription functionality:

- Currently displays "Offline transcription not yet implemented"
- Designed to use local speech recognition models (future implementation)

Dependencies

The project requires the following Python packages:

- `speech_recognition` - For speech recognition functionality
- `socket` - For network connectivity checking (built-in)

Installation

1. Clone the repository:

```
git clone https://github.com/sandrabinoy/Transcriber.git
cd Transcriber
```

2. Install required dependencies:

```
pip install SpeechRecognition
```

3. For microphone support, you may also need:

```
pip install pyaudio
```

Usage

Run the main application:

```
python VoiceRecognition/main.py
```

The application will:

1. Check your internet connection
2. Start the appropriate transcription mode
3. Begin listening for speech
4. Display transcribed text in real-time
5. Stop when you say "exit"

How It Works

1. **Connectivity Check**: The application tests connectivity by attempting to connect to Google's DNS server
2. **Mode Selection**: Based on connectivity, it chooses online (Google API) or offline transcription
3. **Voice Capture**: Uses the system microphone to capture audio
4. **Noise Adjustment**: Automatically adjusts for ambient noise levels
5. **Transcription**: Processes audio and displays the transcribed text
6. **Continuous Loop**: Continues until the user says "exit"

Current Status

Implemented Features ✅

- Internet connectivity detection
- Google Speech Recognition integration
- Real-time voice transcription
- Ambient noise adjustment
- Error handling and user feedback
- Exit command functionality

In Development 🚧

- Offline transcription using Whisper or similar local models
- Audio file transcription support
- Configuration options
- Enhanced error recovery

Development Setup

The project includes a comprehensive `.gitignore` file that excludes:

- Python bytecode files (`__pycache__/`, `*.pyc`)
- Virtual environments
- IDE configuration files
- Audio files and model files
- OS-specific files (`.DS_Store`)

Contributing

This project is set up for collaborative development with:

- Modular architecture for easy feature additions
- Comprehensive error handling
- Clear separation of online/offline functionality
- Git repository configured for the Transcriber project

Future Enhancements

- Implement offline transcription using OpenAI Whisper
- Add support for multiple languages
- Implement audio file transcription
- Add configuration file for customizable settings
- Create GUI interface
- Add voice command recognition beyond transcription

Technical Notes

- Uses Google's DNS server (8.8.8.8) for reliable connectivity testing
- Implements timeout-based connectivity checking (3-second timeout)
- Modular design allows for easy extension and testing
- Error handling covers network issues, audio problems, and recognition failures

---

This project demonstrates a practical implementation of voice recognition technology with intelligent fallback mechanisms for different connectivity scenarios.