🎙️ Whisper Pro Transcriber

Uma aplicação Streamlit avançada para transcrição de áudio usando o modelo Whisper da OpenAI, com funcionalidades profissionais como processamento em lote, gravação ao vivo e interface moderna.

✨ Funcionalidades

🎯 Funcionalidades Principais

- **Transcrição Individual**: Upload e processamento de arquivos únicos
- **Processamento em Lote**: Múltiplos arquivos simultaneamente
- **Gravação ao Vivo**: Captura de áudio diretamente no navegador
- **Múltiplos Formatos**: Suporte a MP3, WAV, M4A, MP4, OGG, FLAC e mais
- **Detecção Automática de Idiomas**: Ou seleção manual entre 25+ idiomas
- **Tradução Automática**: Converta qualquer idioma para inglês

🔧 Funcionalidades Avançadas

- **5 Modelos Whisper**: De tiny (rápido) a large-v2 (máxima precisão)
- **Configurações Avançadas**: Temperatura, beam size, threshold de confiança
- **Presets de Qualidade**: Equilibrado, rápido ou alta qualidade
- **Estimativas de Tempo**: Baseadas no hardware e tamanho do arquivo
- **Otimização GPU/CPU**: Detecção automática e configuração

📊 Saídas e Formatos

- **TXT**: Texto simples
- **SRT**: Legendas para vídeos
- **VTT**: Legendas web
- **JSON**: Dados completos com metadados
- **Timestamps Detalhados**: Por segmento e palavra
- **Análise de Confiança**: Métricas de qualidade da transcrição

🎨 Interface e UX

- **Design Moderno**: Interface responsiva com CSS customizado
- **Modo Escuro/Claro**: Adaptação automática
- **Histórico de Transcrições**: Últimas 50 transcrições salvas
- **Filtros e Busca**: No histórico por data, modelo, etc.
- **Progress Tracking**: Barras de progresso em tempo real

- **Validação Inteligente**: Verificação de arquivos e configurações

🚀 Instalação

Pré-requisitos

- Python 3.8+
- FFmpeg instalado no sistema
- (Opcional) GPU NVIDIA com CUDA para processamento acelerado

1. Clonar o Repositório

bash

```bash
git clone https://github.com/seu-usuario/whisper-pro-transcriber.git
cd whisper-pro-transcriber
```

2. Criar Ambiente Virtual

bash

```bash
python -m venv venv
source venv/bin/activate  # No Windows: venv\Scripts\activate
```

3. Instalar Dependências

bash

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

4. Verificar FFmpeg

bash

```bash
ffmpeg -version
```

Se FFmpeg não estiver instalado:

Ubuntu/Debian:

bash

```bash
sudo apt update
sudo apt install ffmpeg
```

macOS:

```bash
brew install ffmpeg
```

Windows:

- Baixe de https://ffmpeg.org/download.html
- Adicione ao PATH do sistema

5. Executar a Aplicação

```bash
streamlit run main.py
```

A aplicação estará disponível em `http://localhost:8501`

🐳 Docker (Opcional)

Build da Imagem

```bash
docker build -t whisper-transcriber .
```

Executar Container

```bash
docker run -p 8501:8501 whisper-transcriber
```

Docker Compose

```bash
docker-compose up
```

📖 Como Usar

1. Upload de Arquivo Individual

1. Selecione o modelo Whisper desejado na barra lateral
2. Escolha o idioma (ou deixe em detecção automática)
3. Faça upload do arquivo de áudio/vídeo
4. Configure opções avançadas se necessário

5. Clique em "Transcrever Arquivo"

2. Processamento em Lote

1. Vá para a aba "Processamento em Lote"
2. Faça upload de múltiplos arquivos
3. Escolha o formato de saída (individual, consolidado ou ambos)
4. Configure se deve continuar em caso de erro
5. Clique em "Processar Lote"

3. Gravação ao Vivo

1. Vá para a aba "Gravação ao Vivo"
2. Configure qualidade e duração máxima
3. Clique em "Iniciar Gravação"
4. Fale no microfone
5. Clique em "Parar Gravação" e depois "Transcrever"

4. Histórico

- Visualize transcrições anteriores na aba "Histórico"
- Filtre por data, modelo ou outros critérios
- Baixe transcrições antigas em qualquer formato

⚙️ Configurações Avançadas

Modelos Whisper

Modelo	Tamanho	Velocidade	Precisão	Uso Recomendado
tiny	~39MB	⚡⚡⚡⚡⚡	★★	Testes rápidos
base	~74MB	⚡⚡⚡⚡	★★★	Uso geral
small	~244MB	⚡⚡⚡	★★★★	Melhor equilíbrio
medium	~769MB	⚡⚡	★★★★★	Alta precisão
large-v2	~1550MB	⚡	★★★★★	Máxima precisão

Presets de Qualidade

- **Rápido**: Processamento mais veloz, menor precisão
- **Equilibrado**: Melhor relação velocidade/qualidade
- **Alta Qualidade**: Máxima precisão, processamento mais lento

Configurações de Temperatura

- **0.0**: Determinístico, sempre o mesmo resultado
- **0.1-0.3**: Pouca variação, boa para áudio claro
- **0.4-0.7**: Variação moderada, útil para áudio com ruído
- **0.8-1.0**: Alta variação, experimental

🔧 Solução de Problemas

Problemas Comuns

1. "FFmpeg não encontrado"

```bash
# Instalar FFmpeg conforme instruções acima
# Verificar se está no PATH
echo $PATH  # Linux/Mac
echo %PATH% # Windows
```

2. "Erro de memória GPU"

- Use um modelo menor (tiny, base, small)
- Reduza o tamanho do arquivo de áudio
- Feche outras aplicações que usam GPU

3. "Arquivo muito grande"

- Comprima o áudio antes do upload
- Use processamento em lote para arquivos múltiplos
- Configure limite máximo de arquivo

4. "Transcrição imprecisa"

- Use modelo maior (medium, large-v2)
- Melhore a qualidade do áudio
- Ajuste configurações avançadas
- Especifique o idioma correto

Logs e Debug

bash

```bash
# Executar com logs detalhados
streamlit run main.py --logger.level=debug

# Verificar dependências
pip list | grep -E "(streamlit|torch|whisper)"
```

🚀 Performance e Otimizações

GPU vs CPU

- **GPU (CUDA)**: 3-5x mais rápido que CPU
- **CPU**: Funciona em qualquer sistema, mas mais lento
- **Recomendação**: GPU com 4GB+ VRAM para modelos grandes

Tamanhos de Arquivo Recomendados

Hardware	Arquivo Individual	Lote Total
CPU básico	< 50MB	< 200MB
CPU potente	< 200MB	< 1GB
GPU 4GB	< 500MB	< 2GB
GPU 8GB+	< 2GB	< 10GB

Dicas de Performance

1. **Use o modelo apropriado** para seu hardware
2. **Processe em lote** para múltiplos arquivos
3. **Comprima áudios longos** antes do upload
4. **Feche outras aplicações** que usam GPU/CPU intensivamente
5. **Use SSD** para arquivos temporários

🔒 Privacidade e Segurança

- **100% Local**: Todo processamento é feito no seu computador
- **Sem Upload**: Seus arquivos nunca saem da sua máquina
- **Arquivos Temporários**: Automaticamente limpos após processamento
- **Sem Logging**: Transcrições não são salvas permanentemente (a menos que você baixe)

🛠️ Desenvolvimento

Estrutura do Projeto

```
whisper-pro-transcriber/
├── main.py                 # Interface principal Streamlit
├── transcriber.py          # Lógica de transcrição Whisper
├── utils.py                # Utilitários e validações
├── audio_recorder.py       # Funcionalidade de gravação
├── requirements.txt        # Dependências Python
├── Dockerfile              # Container Docker
├── docker-compose.yml      # Orquestração Docker
└── README.md               # Esta documentação
```

Contribuindo

1. Fork o repositório
2. Crie uma branch para sua feature
3. Commit suas mudanças
4. Abra um Pull Request

Extensões Futuras

- [ ] Diarização de falantes (speaker separation)
- [ ] Integração com YouTube/URLs
- [ ] API REST para integração
- [ ] Suporte a mais formatos de saída
- [ ] Interface multi-idioma
- [ ] Análise de sentimentos
- [ ] Resumo automático com LLM

📄 Licença

Este projeto está licenciado sob a MIT License - veja o arquivo LICENSE para detalhes.

🙏 Agradecimentos

- OpenAI Whisper - Modelo de transcrição
- Streamlit - Framework web
- FFmpeg - Processamento de mídia

📞 Suporte

- **Issues**: GitHub Issues
- **Documentação**: Este README e comentários no código
- **Comunidade**: Discussões no GitHub

Desenvolvido com 💔 usando OpenAI Whisper e Streamlit