🎧 PDF to Audio Converter

A complete full-stack web application that converts PDF documents to high-quality audio files using MiniMax TTS (Text-to-Speech) technology. Features a beautiful React frontend and robust FastAPI backend.

✨ Live Demo

🚀 **Frontend**: https://yr2ynu8au2.space.minimax.io

PDF to Audio App

🌟 Features

Frontend (React + TypeScript + TailwindCSS)

- 📱 **Responsive Design**: Beautiful, mobile-first UI that works on all devices
- 🎯 **Drag & Drop Upload**: Intuitive PDF file upload with visual feedback
- 📊 **Real-time Progress**: Live progress tracking during conversion
- 🎵 **Audio Player**: Built-in HTML5 audio player with controls
- ⬇️ **Download Support**: Easy audio file download functionality
- 🎨 **Modern UI**: Clean, elegant design with smooth animations

Backend (FastAPI + Python)

- 📄 **Advanced PDF Processing**: Multi-library text extraction (pdfplumber, PyMuPDF)
- 🎤 **High-Quality TTS**: MiniMax AI-powered text-to-speech conversion
- 🔧 **Intelligent Chunking**: Smart text splitting for optimal TTS processing
- 🎵 **Audio Processing**: Automatic audio merging and optimization

- ⚡ **Async Processing**: Non-blocking background task processing
- 🔐 **Secure & Robust**: Comprehensive error handling and validation

🏗️ Architecture

```
┌──────────────────┐    ┌──────────────────┐    ┌──────────────────┐
│   React Frontend│    │  FastAPI Backend│    │   MiniMax TTS   │
│                  │    │                  │    │       API        │
│  • File Upload  │──▶│  • PDF Parser   │──▶│  • Voice Synth  │
│  • Progress UI  │◀──│  • Text Chunker │◀──│  • Audio Gen    │
│  • Audio Player │    │  • Audio Merger │    │                  │
└──────────────────┘    └──────────────────┘    └──────────────────┘
```

🚀 Quick Start

Prerequisites

- Node.js 18+ and pnpm
- Python 3.11+
- MiniMax API account and credentials

1. Clone Repository

```
git clone <repository-url>
cd pdf-to-audio-converter
```

2. Frontend Setup

```
cd pdf-to-audio-frontend
pnpm install
pnpm dev
```

3. Backend Setup

```
cd pdf-to-audio-backend
pip install -r requirements.txt

# Configure environment
cp .env.example .env
# Edit .env with your MiniMax API credentials

# Run development server
python dev.py
```

📁 Project Structure

```
pdf-to-audio-converter/
├── 📁 pdf-to-audio-frontend/     # React frontend application
│   ├── 📁 src/
│   │   ├── 📁 components/        # React components
│   │   │   ├── Header.tsx        # App header with navigation
│   │   │   ├── Footer.tsx        # App footer
│   │   │   ├── FileUploader.tsx  # Drag & drop upload
│   │   │   ├── ConversionProgress.tsx # Progress indicator
│   │   │   └── AudioPlayer.tsx   # Audio playback component
│   │   ├── App.tsx              # Main app component
│   │   └── main.tsx             # App entry point
│   ├── 📁 public/               # Static assets
│   ├── package.json             # Dependencies
│   ├── vite.config.ts           # Vite configuration
│   ├── vercel.json              # Vercel deployment config
│   └── .env.example             # Environment template
│
├── 📁 pdf-to-audio-backend/      # FastAPI backend
│   ├── main.py                  # FastAPI application
│   ├── pdf_processor.py         # PDF text extraction
│   ├── tts_service.py           # MiniMax TTS integration
│   ├── audio_utils.py           # Audio processing utilities
│   ├── config.py                # Configuration management
│   ├── requirements.txt         # Python dependencies
│   ├── Dockerfile               # Docker configuration
│   ├── render.yaml              # Render deployment
│   ├── railway.json             # Railway deployment
│   └── .env.example             # Environment template
│
└── README.md                    # This file
```

🔧 Configuration

Frontend Environment Variables

```
# .env
VITE_API_URL=http://localhost:8000  # Backend API URL
```

Backend Environment Variables

```
# .env
MINIMAX_API_KEY=your_api_key_here
MINIMAX_GROUP_ID=your_group_id_here
DEBUG=True
PORT=8000
MAX_FILE_SIZE_MB=50
```

🚀 Deployment

Frontend (Vercel)

1. **Connect Repository**: Link your GitHub repository to Vercel
2. **Configure Environment**: Set `VITE_API_URL` to your backend URL
3. **Deploy**: Automatic deployment on push to main branch

▲ Deploy

Backend (Render)

1. **Create Web Service**: Connect your repository on Render
2. **Environment Variables**: Configure API keys and settings

3. **Auto Deploy**: Uses included `render.yaml` configuration

Deploy to Render

Backend (Railway)

1. **Connect Repository**: Link GitHub repository to Railway
2. **Environment Setup**: Configure MiniMax API credentials
3. **Deploy**: Automatic deployment with `railway.json`

Deploy on Railway

Docker Deployment

```
# Backend
cd pdf-to-audio-backend
docker build -t pdf-audio-backend .
docker run -p 8000:8000 --env-file .env pdf-audio-backend

# Frontend (with nginx)
cd pdf-to-audio-frontend
docker build -t pdf-audio-frontend .
docker run -p 3000:80 pdf-audio-frontend
```

🎨 Design Philosophy

This application follows modern UI/UX principles:

- **Minimalist Elegance**: Clean, uncluttered interface focused on functionality
- **Visual Hierarchy**: Clear information structure guiding user attention
- **Responsive Design**: Seamless experience across all device sizes
- **Accessibility**: Inclusive design with proper ARIA labels and keyboard navigation

- **Performance**: Optimized loading and smooth interactions

Color Palette

- **Primary**: Indigo (#4F46E5) - Trust and technology
- **Secondary**: Slate (#64748B) - Professional and readable
- **Accent**: Green (#10B981) - Success and completion
- **Background**: Gradient from Slate to Indigo - Modern depth

🔄 Processing Pipeline

1. 📤 **File Upload**: User uploads PDF via drag & drop interface
2. ✅ **Validation**: File type and size validation
3. 📄 **Text Extraction**: Advanced PDF parsing with fallback methods
4. ✂️ **Text Chunking**: Intelligent text splitting for TTS optimization
5. 🎤 **TTS Conversion**: MiniMax API converts text chunks to audio
6. 🎵 **Audio Merging**: Combine audio chunks into final MP3 file
7. ⬇️ **Download Ready**: Audio available for playback and download

🛠️ Development

Adding New Features

Frontend Components

```
# Create new component
touch src/components/NewComponent.tsx
```

Backend Endpoints

```
# Add to main.py
@app.post("/api/new-endpoint")
async def new_endpoint():
    pass
```

TTS Voice Options

```
# Available voices in tts_service.py
voices = {
    "female-qn-qingse": "Clear female voice",
    "male-qn-qingse": "Clear male voice",
    "female-shaonv": "Young female voice",
    "male-youthful": "Youthful male voice"
}
```

Testing

Frontend

```
cd pdf-to-audio-frontend
pnpm dev          # Development server
pnpm build        # Production build
pnpm preview      # Preview production build
```

Backend

```
cd pdf-to-audio-backend
python dev.py     # Development server
uvicorn main:app --reload  # Alternative startup
```

📊 Performance & Scalability

Current Limitations

- **File Size**: 50MB PDF limit
- **Processing Time**: 1-3 minutes depending on document length
- **Concurrent Users**: Single-instance backend (use load balancer for scale)

Optimization Opportunities

- **Caching**: Redis for conversion status and results
- **Queue System**: Celery for background task processing
- **CDN**: Static asset delivery optimization
- **Database**: PostgreSQL for persistent job storage

🐛 Troubleshooting

Common Issues

Frontend

- **Build Errors**: Check Node.js version (18+ required)
- **API Connection**: Verify backend URL in environment variables
- **Upload Issues**: Check file size and type restrictions

Backend

- **TTS API Errors**: Verify MiniMax API credentials and rate limits
- **PDF Processing**: Try different PDF files, check for text content
- **Audio Merge Failed**: Install ffmpeg for better audio processing

Debug Mode

```
# Frontend
VITE_DEBUG=true pnpm dev

# Backend
DEBUG=true python dev.py
```

🤝 Contributing

We welcome contributions! Please follow these steps:

1. **Fork** the repository
2. **Create** feature branch: `git checkout -b feature/amazing-feature`
3. **Commit** changes: `git commit -m 'Add amazing feature'`
4. **Push** branch: `git push origin feature/amazing-feature`
5. **Open** Pull Request

Code Style

- **Frontend**: ESLint + Prettier configuration
- **Backend**: Black + isort formatting
- **Commits**: Conventional Commits format

📄 License

This project is licensed under the MIT License - see the LICENSE file for details.

🙏 Acknowledgments

- **MiniMax**: Advanced TTS API technology
- **React**: Frontend framework and ecosystem
- **FastAPI**: High-performance Python web framework
- **TailwindCSS**: Utility-first CSS framework
- **Vercel**: Frontend hosting and deployment
- **Render**: Backend hosting and deployment

📞 Support

For support and questions:

- 📧 **Email**: support@audiopdface.com
- 🐛 **Issues**: GitHub Issues
- 💬 **Discussions**: GitHub Discussions
- 📖 **Documentation**: Project Wiki

🎧 Turn your PDFs into podcasts with AI-powered voice synthesis

Made with ❤️ using React, FastAPI, and MiniMax TTS