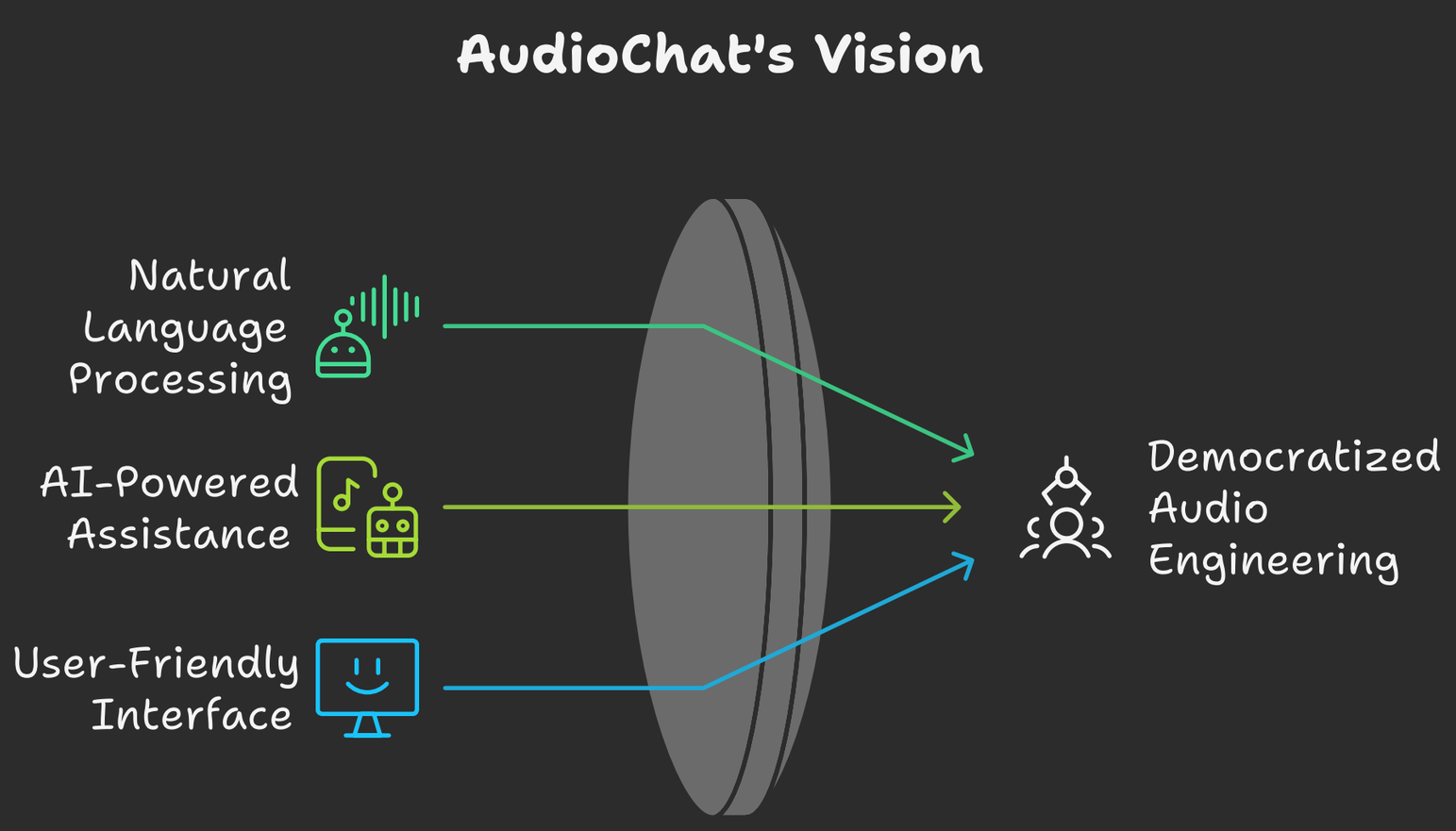
AudioChat: AI-Powered Audio Engineering Assistant

AudioChat is an innovative AI-powered audio engineering assistant designed to simplify the complexities of audio editing, mixing, and mastering. By leveraging natural language processing, AudioChat allows users to make precise audio modifications simply by describing the desired changes in plain language. This document outlines the features, functionality, technology stack, setup instructions, usage guidelines, project structure, configuration details, and contribution guidelines for AudioChat. Whether you're a seasoned audio professional or a beginner, AudioChat aims to democratize audio engineering, making it accessible and intuitive for everyone.

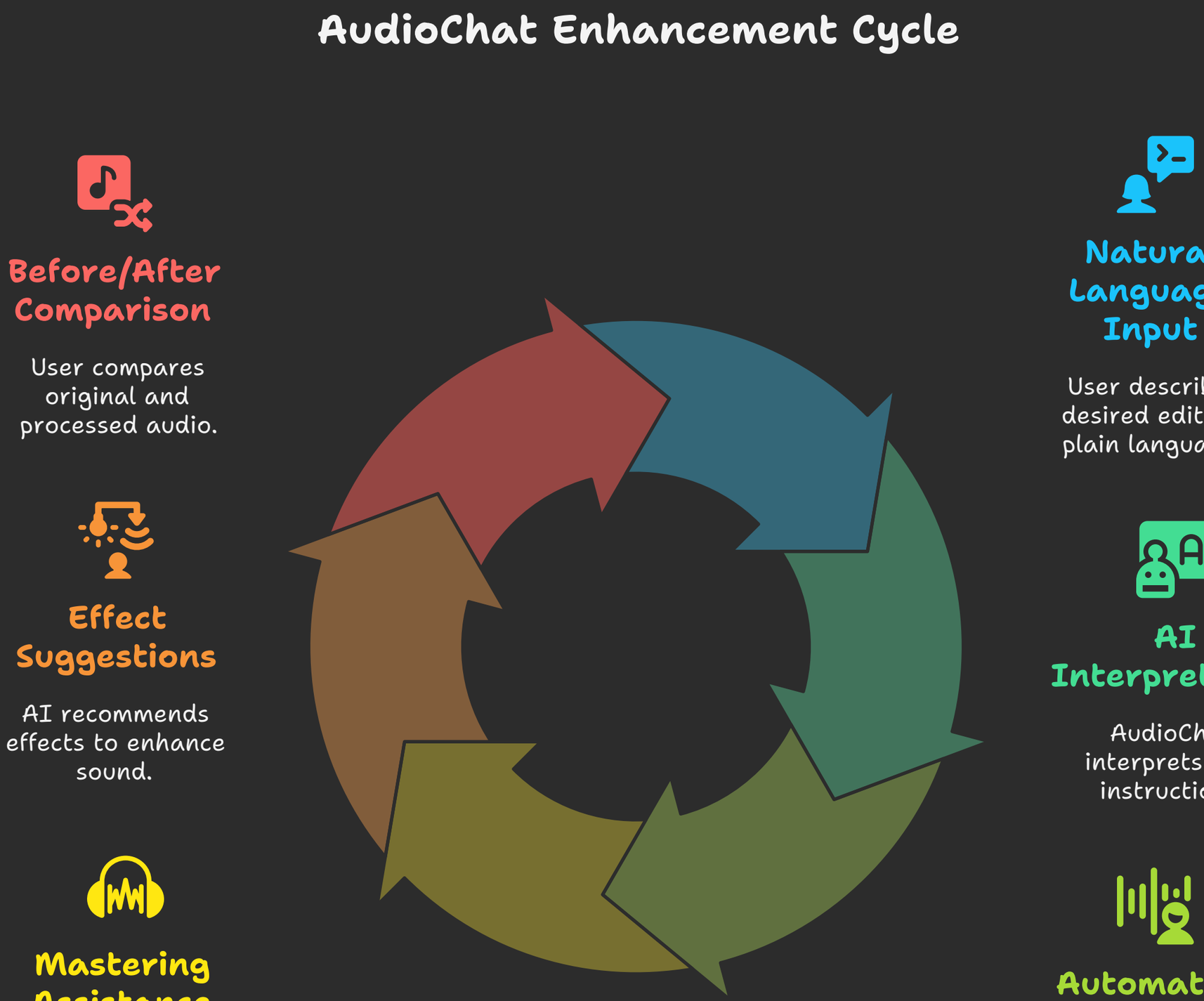


Features

Audio Engineering Capabilities

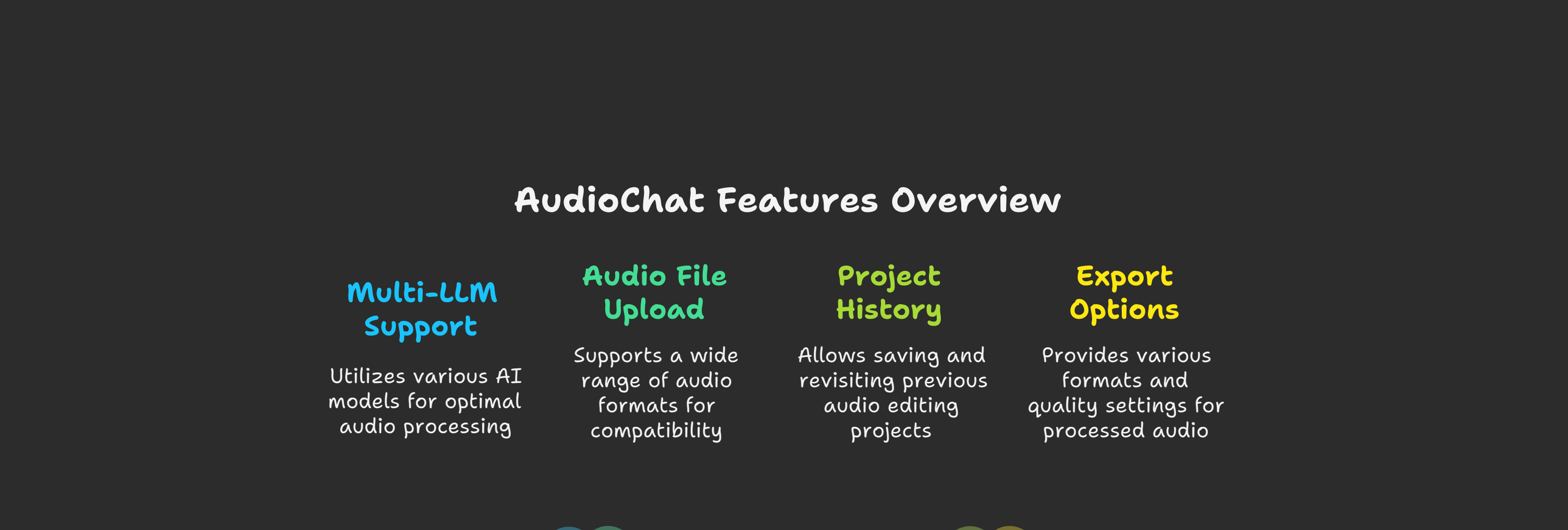
- **Natural Language Audio Editing**: Describe audio edits in plain language (e.g., "Make the vocals louder" or "Add more bass"). AudioChat interprets these instructions and applies the necessary audio processing techniques.
- **Automated Mixing**: Apply professional mixing techniques through AI-guided processing. AudioChat can automatically balance levels, EQ frequencies, and add effects to create a polished mix.
- **Mastering Assistant**: Get AI assistance for finalizing and polishing your tracks. AudioChat can optimize the overall loudness, clarity, and dynamic range of your audio.
- **Audio Effect Suggestions**: Receive recommendations for effects and processing chains. AudioChat analyzes your audio and suggests appropriate effects to enhance the sound.

- **Before/After Comparison**: Compare original and processed audio files. Easily switch between the original and processed versions to evaluate the changes.



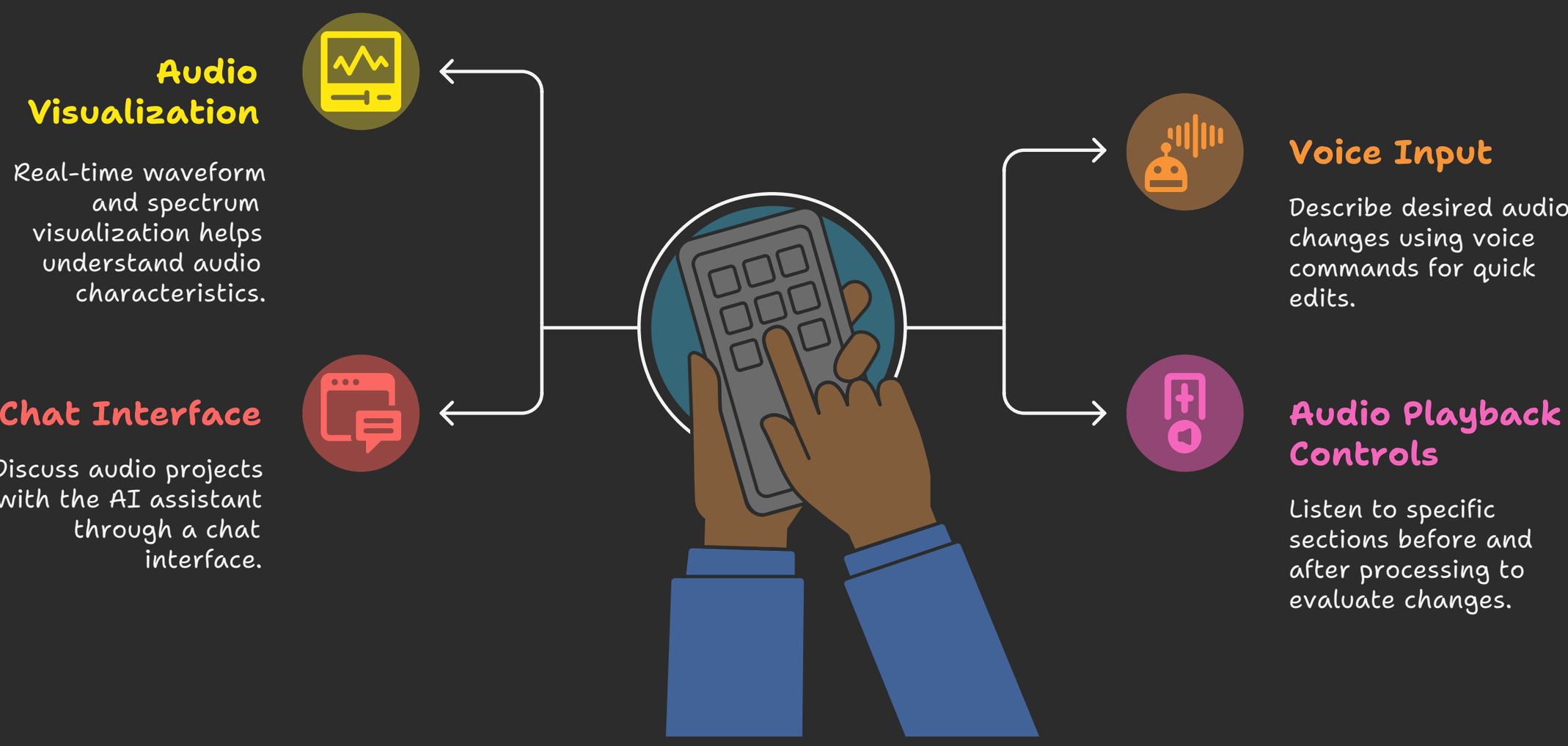
Core Functionality

- **Multi-LLM Support**: Leverage different AI models specialized in audio engineering knowledge. AudioChat can utilize various language models to provide the best possible audio processing.
- **Audio File Upload**: Upload WAV, MP3, FLAC, and other common audio formats. AudioChat supports a wide range of audio formats for maximum compatibility.
- **Project History**: Save and revisit previous audio editing projects. Keep track of your audio projects and easily access them later.
- **Export Options**: Download processed audio in various formats and quality settings. Export your audio in the format and quality you need.



User Experience

- **Audio Visualization**: Real-time waveform and spectrum visualization. Visualize your audio to better understand its characteristics.
- **Voice Input**: Describe desired audio changes using your voice. Use voice commands to quickly and easily make audio edits.
- **Chat Interface**: Discuss your audio project with the AI assistant. Interact with the AI assistant through a chat interface to refine your audio.
- **Audio Playback Controls**: Listen to specific sections before and after processing. Precisely evaluate the changes made to your audio.



Interface & Design

- **Dark Theme**: Professional studio-like dark interface optimized for audio work. Reduce eye strain and improve focus with a dark theme.
- **Audio Processing Settings**: Fine-tune default processing parameters. Customize the audio processing to your specific needs.
- **API Key Management**: Securely store and manage API keys for different providers. Keep your API keys safe and organized.
- **Responsive Design**: Works well on various screen sizes, from studio monitors to tablets. Use AudioChat on any device, anywhere.

AudioChat Features

Dark Theme

Professional dark interface optimized for audio work. Reduce eye strain and improve focus.

Audio Processing

Fine-tune default processing parameters to your specific needs. Customize audio processing.

API Key Management

Securely store and manage API keys for different providers. Keep API keys safe.

Responsive Design

Works well on various screen sizes, from studio monitors to tablets. Use AudioChat anywhere.

Technology Stack

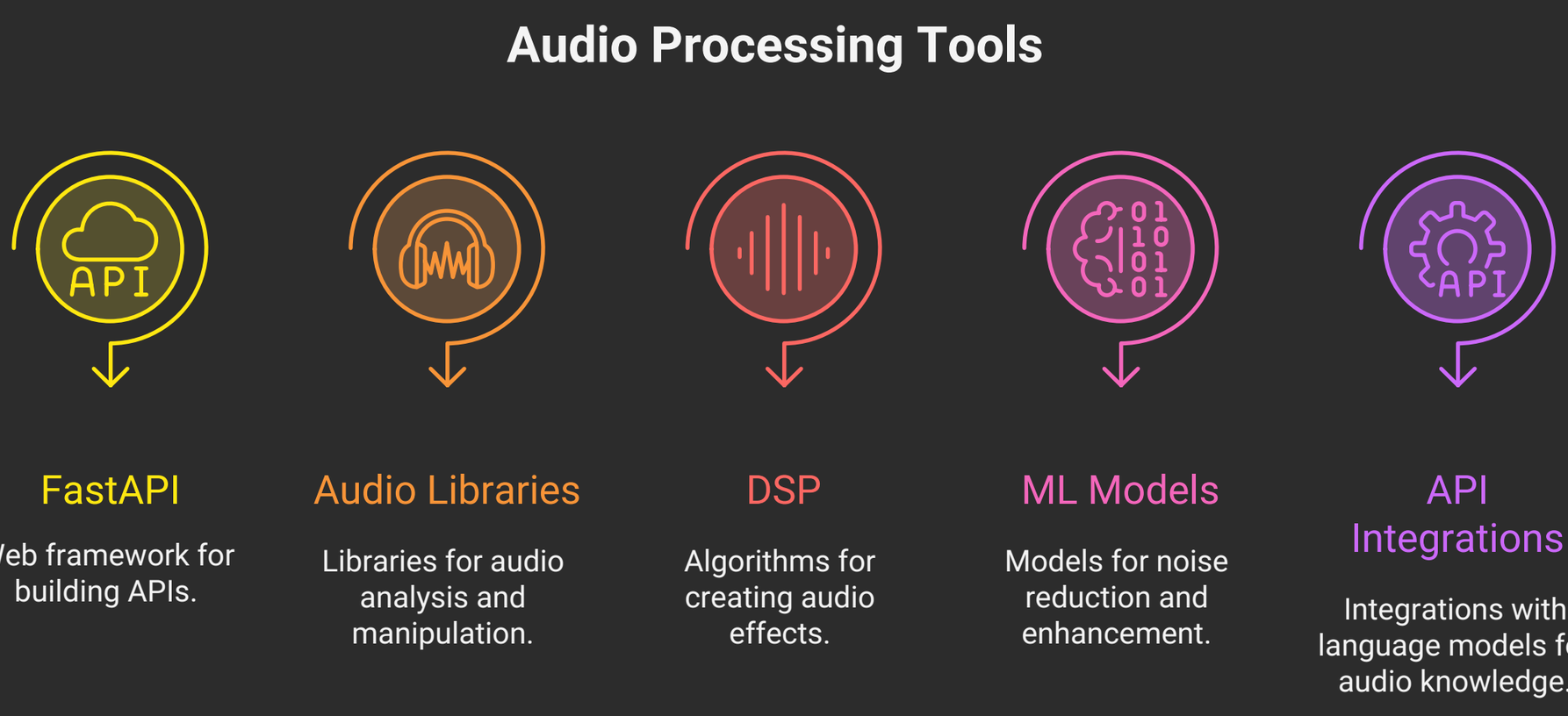
Frontend

- React.js: A JavaScript library for building user interfaces.
- Context API for state management: A way to manage state in React applications.
- CSS for styling: Used for styling the user interface.
- Web Audio API for audio visualization: A JavaScript API for processing and synthesizing audio in web applications.

AudioChat Technology Stack Overview

Backend

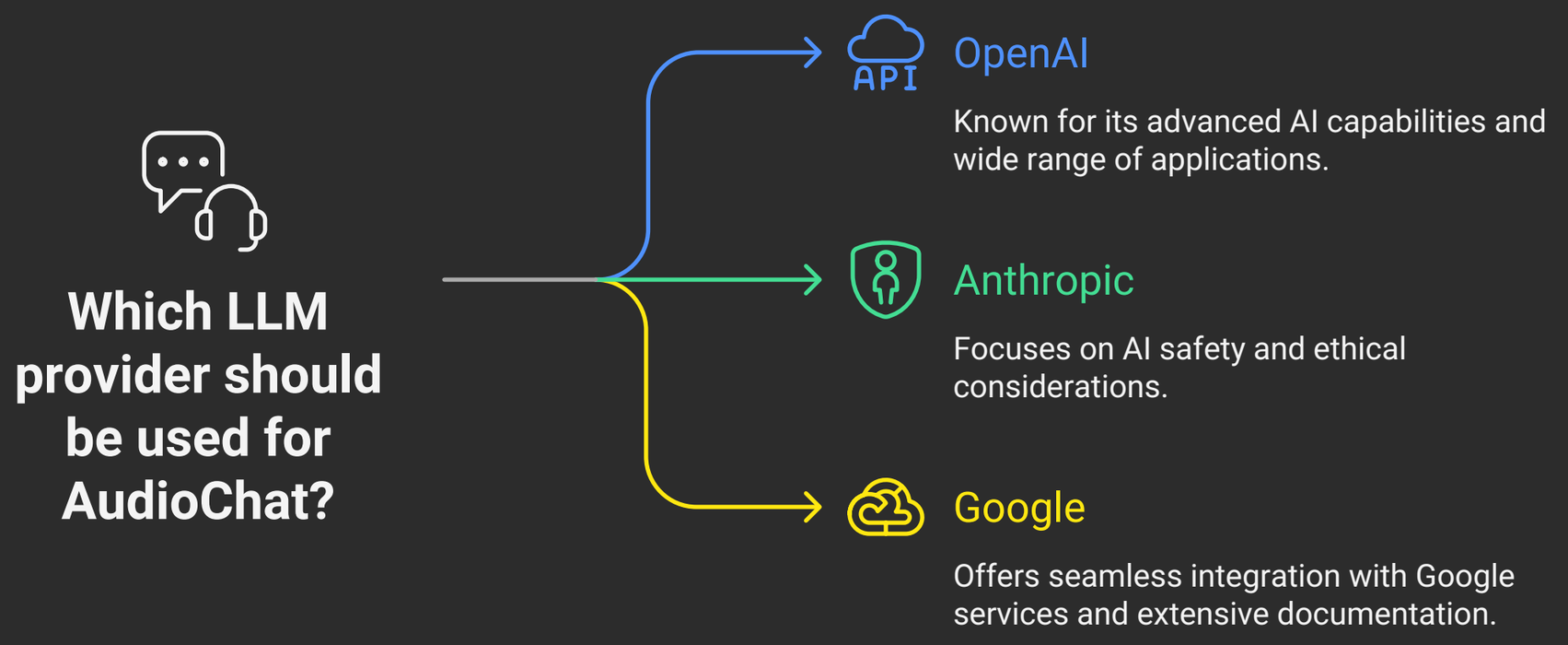
- Python FastAPI: A modern, fast (high-performance), web framework for building APIs with Python 3.7+.
- Audio processing libraries (librosa, pydub, SoundFile): Libraries for audio analysis and manipulation.
- Digital Signal Processing (DSP) for audio effects: Algorithms for creating audio effects.
- Machine learning models for intelligent audio processing: Models for tasks such as noise reduction and audio enhancement.
- API integrations with various LLM providers for audio engineering knowledge: Integrations with language models for understanding audio engineering concepts.



Getting Started

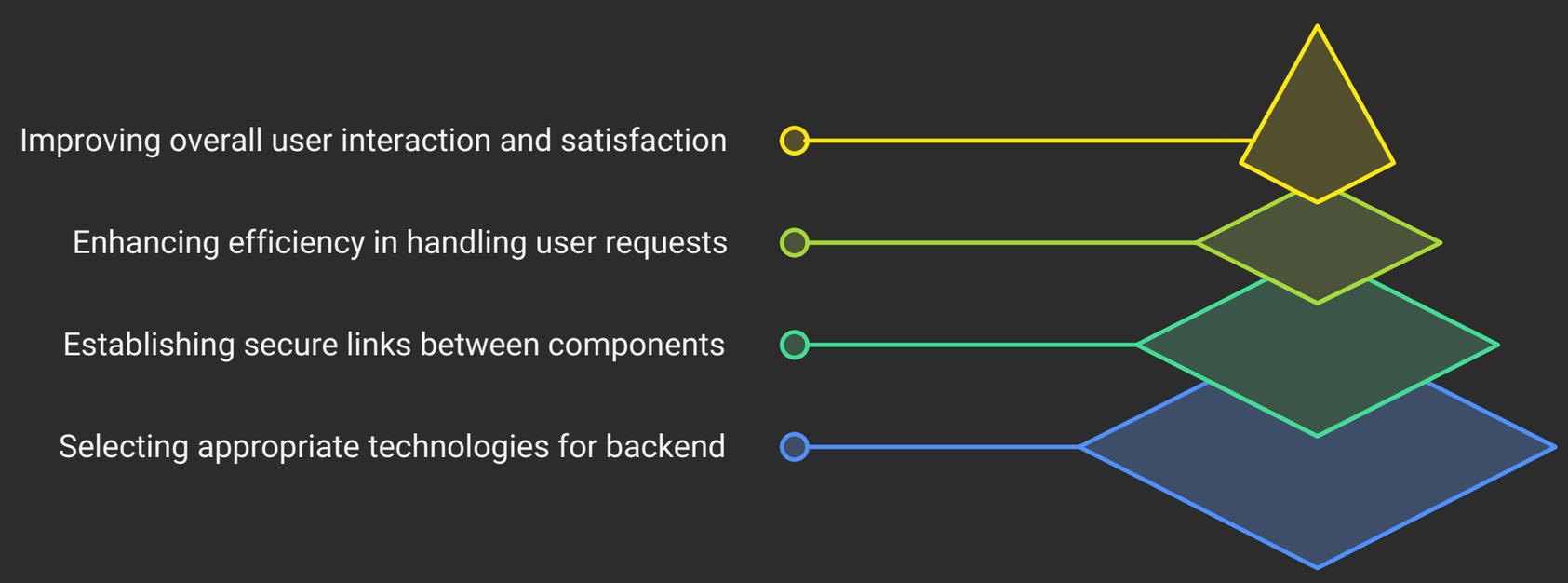
Prerequisites

- Node.js (v14 or later)
- Python (v3.8 or later)
- API keys for the LLM providers you want to use (OpenAI, Anthropic, Google)



Installation

Backend Setup



1. Navigate to the backend directory:

```
cd backend
```

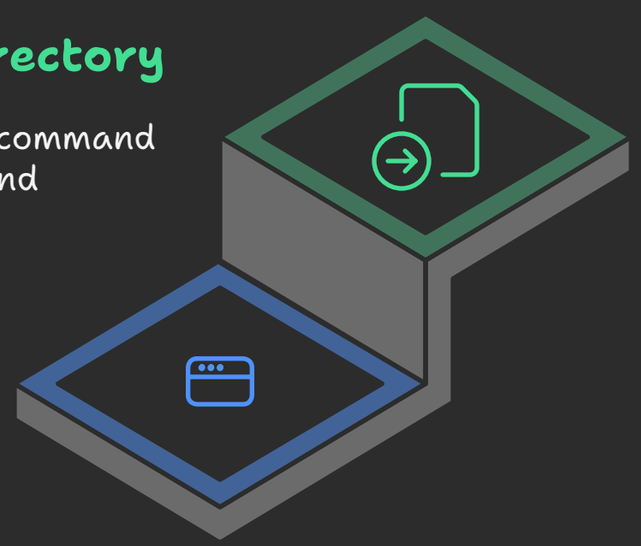
Setting Up Backend Directory

2 **Navigate to Directory**

Use the 'cd backend' command to move to the backend directory.

1 **Open Terminal**

Initiate the process by opening your terminal.



2. Run the setup script to create a virtual environment and install dependencies:

```
setup.bat  # On Windows
./setup.sh  # On Linux/Mac
```

3. Create a .env file based on the .env.example template and add your API keys:

```
cp .env.example .env
```

4. Start the backend server:

```
run.bat  # On Windows
./run.sh  # On Linux/Mac
```

Frontend Setup

1. Install dependencies:

```
npm install
```

2. Start the development server:

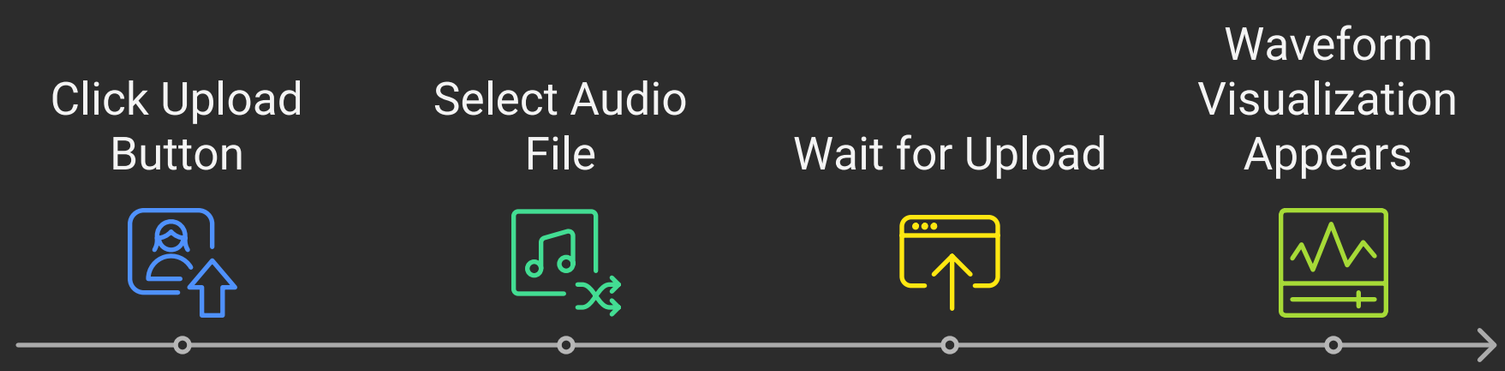
```
npm start
```

3. Open your browser and navigate to http://localhost:3000

Usage

Uploading Audio Files

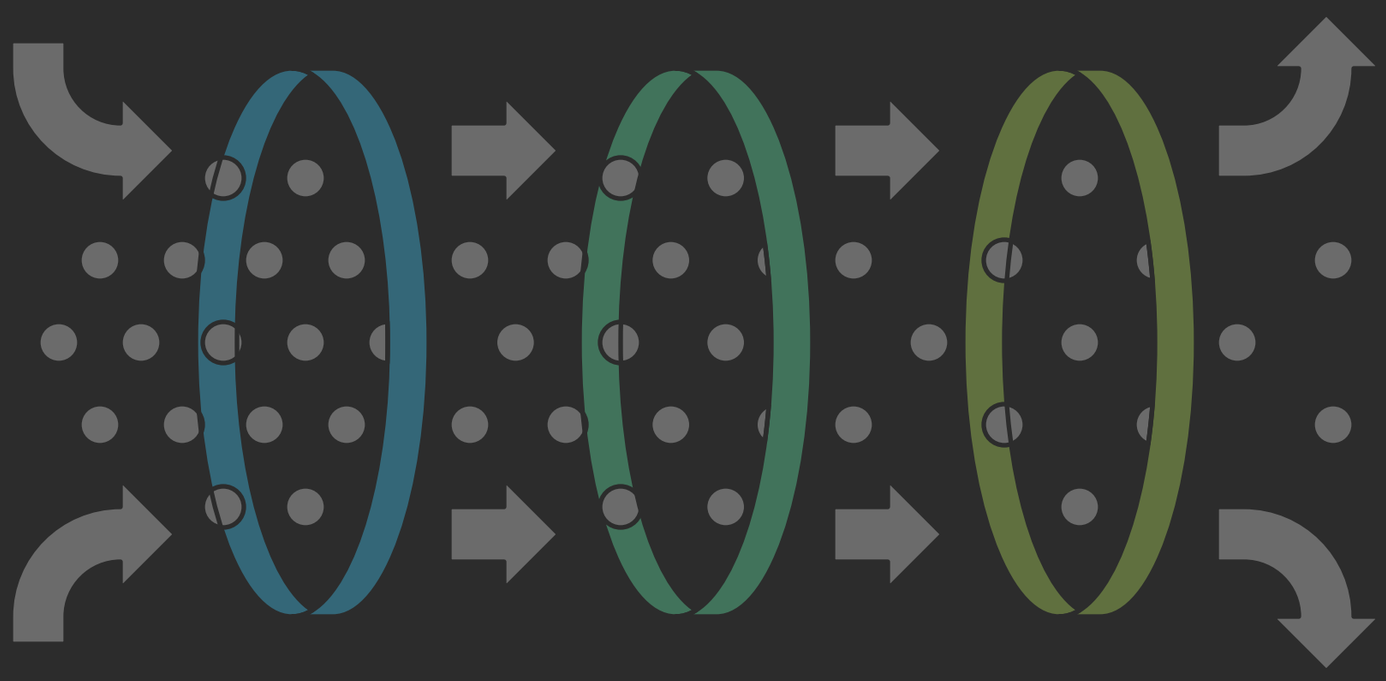
1. Click the "Upload" button in the chat interface
2. Select an audio file from your computer (WAV, MP3, FLAC, etc.)
3. Wait for the file to upload and process
4. The waveform visualization will appear once processing is complete



Requesting Audio Edits

1. Type or speak your desired audio edits in natural language
    - Example: "Reduce the background noise and make the vocals clearer"
    - Example: "Add a subtle reverb to the guitar and boost the low end"
2. The AI will analyze your request and apply appropriate audio processing
3. You'll receive the processed audio file with a description of changes made

Audio Editing Process Funnel



AI Analysis

AI interprets the user's request

Audio Processing

AI applies necessary audio edits

Output Delivery

User receives the edited audio file

Comparing Before/After

1. Use the audio player controls to listen to the original file
2. Switch to the processed version using the toggle button
3. A/B test specific sections to hear the differences

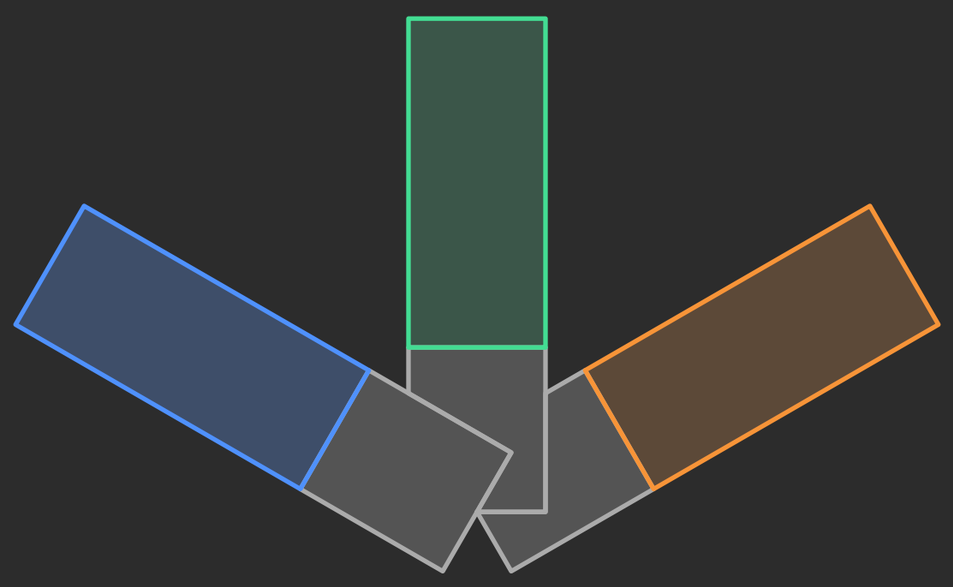
How to compare audio versions?

Switch to Processed

Toggle to the processed version for comparison.

Listen to Original

Use audio player controls to hear the original file.



A/B Test Sections

Compare specific sections to identify differences.

Fine-tuning Processing

1. If the results aren't exactly what you wanted, provide more specific feedback
    - Example: "That's good, but can you make the reverb more subtle?"
2. The AI will apply incremental changes based on your feedback
3. Continue refining until you're satisfied with the results

Audio Refinement Process

Refine Results

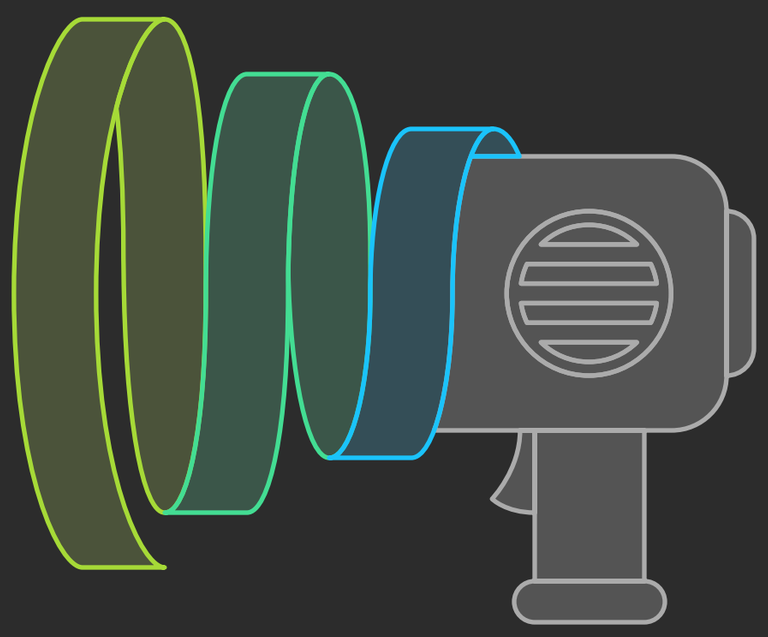
Users continue refining until satisfied with the audio

Apply Changes

The AI implements incremental changes based on feedback

Provide Feedback

Users offer specific feedback on the audio edit



Saving and Exporting

1. Once you're happy with the processed audio, click "Save Project"
2. Choose your preferred export format and quality settings
3. Download the final audio file to your computer

Exporting Processed Audio

3 **Download File**

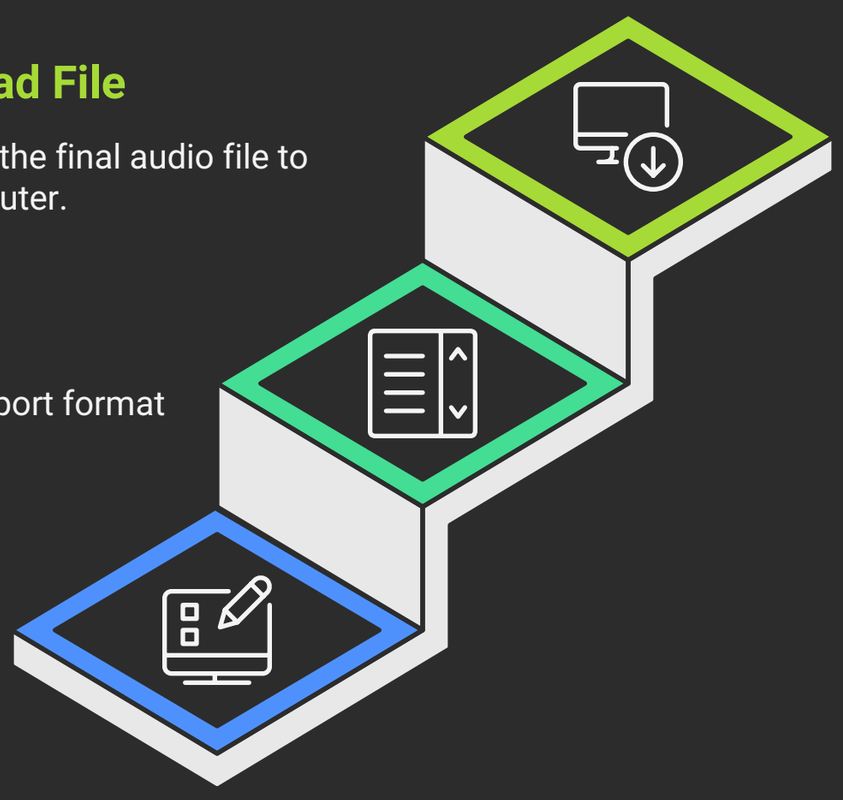
Download the final audio file to your computer.

2 **Choose Format**

Select your preferred export format and quality settings.

1 **Save Project**

Click "Save Project" to preserve your work.



Project Structure

```
audiochat/
├── backend/              # Python FastAPI backend
│   ├── env/              # Python virtual environment
│   ├── main.py           # Main API entry point
│   └── requirements.txt  # Python dependencies
├── public/               # Static assets
└── src/                  # React frontend
    ├── components/       # UI components
    │   ├── AudioRecorder.js      # Voice recording component
    │   ├── AudioVisualizer.js    # Audio waveform visualization
    │   ├── ChatInterface.js      # Main chat interface
    │   ├── MessageList.js        # Chat message display
    │   └── VoiceSelector.js      # Voice selection component
    ├── context/          # React context providers
    │   └── SettingsContext.js    # Global settings management
    ├── services/         # API service functions
```

Configuration

Environment Variables

Create a .env file in the backend directory with the following variables:

```
OPENAI_API_KEY=your_openai_api_key_here
ANTHROPIC_API_KEY=your_anthropic_api_key_here
GOOGLE_API_KEY=your_google_api_key_here
```

API Keys

API keys can also be managed through the application interface:

1. Go to Settings > API Keys
2. Enter your API keys for each provider
3. Click "Save Keys"

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

1. Fork the repository
2. Create your feature branch (git checkout -b feature/amazing-feature)
3. Commit your changes (git commit -m 'Add some amazing feature')
4. Push to the branch (git push origin feature/amazing-feature)
5. Open a Pull Request

License

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

Acknowledgments

- OpenAI for GPT models and Whisper API
- Anthropic for Claude models
- Google for Gemini models
- The React and FastAPI communities for excellent documentation and tools