DJ App: Technical Specification Document

Project Specification

December 2025

Contents

1 Overview **2**

2 Objectives **2**

3 User Roles **2**

4 Core Features (MVP) **2**

5 System Architecture **2**
5.1 Backend (C++) 2
5.2 Audio Engine (C++) 3
5.3 Frontend (Web or Desktop) 3

6 Technology Stack **3**
6.1 Programming Languages 3
6.2 Libraries 3
6.3 External Services 3

7 Legal Considerations **3**

8 Minimum Viable Product (MVP) **4**

9 Future Additions (What To Add) **4**

10 Milestones **4**
10.1 Phase 1 (Basic) 4
10.2 Phase 2 (Mixing) 4
10.3 Phase 3 (Pro) 4

1 Overview

The DJ Application enables users to search, load and mix audio tracks sourced from YouTube. The core audio processing and mixing engine is implemented in C++, while a frontend client (web or desktop) provides controls for playback, crossfading, and browsing.

The application focuses on legal audio streaming, low-latency mixing, and extensibility for future audio manipulation features.

2 Objectives

- Search and stream music using the YouTube Data API.
- Perform real-time audio playback.
- Mix two audio sources simultaneously.
- Provide crossfade and volume control.

3 User Roles

User Type	Capabilities
Guest	Search and play available tracks
DJ (default)	Mix tracks, manage audio engine, control playback

4 Core Features (MVP)

- YouTube search (API v3)
- Metadata retrieval (title, thumbnails, duration)
- Streaming playback (no downloads)
- Dual-deck audio engine (Track A and Track B)
- Crossfade slider between decks
- Volume and seek control

5 System Architecture

5.1 Backend (C++)

- HTTP client module for YouTube API requests
- Authentication module (API key management)
- JSON parsing and result formatting
- REST or gRPC endpoint exposure

5.2 Audio Engine (C++)

- Low-latency audio playback
- Dual-channel mixing
- Real-time crossfade handler
- Buffer-level volume control

5.3 Frontend (Web or Desktop)

- Search bar (YouTube query)
- Dual deck interface
- Crossfade slider
- Playback buttons (play, pause, stop)

6 Technology Stack

6.1 Programming Languages

- C++ (primary)

6.2 Libraries

- PortAudio / JUCE (audio)
- cpp-httplib / Boost.Beast (HTTP)
- nlohmann::json (JSON parsing)

6.3 External Services

- YouTube Data API v3

7 Legal Considerations

- Audio must be streamed, not downloaded.
- No permanent storage of audio files.
- Stick to YouTube TOS-compliant playback.

8 Minimum Viable Product (MVP)

1. Console-based YouTube search (returns metadata)
2. Play audio stream
3. Display track metadata
4. Dual playback and manual crossfade

9 Future Additions (What To Add)

- BPM detection and auto beat matching
- Equalizer controls (bass, mid, treble)
- Waveform visualization using FFT
- Effects (delay, reverb, filters)
- Live playlist sharing
- Remote listeners / streaming
- AI-generated playlist recommendations
- Mobile client

10 Milestones

10.1 Phase 1 (Basic)

- Search and metadata
- Single audio playback

10.2 Phase 2 (Mixing)

- Dual decks
- Crossfade engine

10.3 Phase 3 (Pro)

- BPM sync
- EQ
- Waveform UI