DesiDJ – AI-Assisted Continuous-Mix Platform for Personalized Music Playback and Real-Time Audio Transitions

1. Cross-Reference to Related Applications

None.

2. Field of the Invention

The present invention relates to computer-implemented audio processing systems, and more particularly to methods and systems that generate continuous, beat-matched music streams with dynamic emotion-based playlist curation.

3. Background of the Invention

- Traditional music-streaming services deliver discrete tracks separated by silence or abrupt changes.
- Professional DJs manually select songs, match tempos, equalize energy levels, and craft smooth transitions.
- Consumers lack tools that autonomously emulate DJ-style mixes based on individual emotional context and streaming-service libraries.

There is therefore a need for an automated platform that (i) analyses user-specific emotional cues, (ii) curates playlists from cloud libraries such as Spotify®, and (iii) renders seamless track-to-track transitions in real time without manual DJ intervention.

4. Summary of the Invention

DesiDJ provides: a) An emotion-analysis module that classifies text input (e.g., social-media posts or chat messages) with cloud-based natural-language services (e.g., IBM Watson®) to derive mood vectors. b) A playlist-curation engine that queries third-party music APIs (e.g., Spotify) to retrieve candidate tracks whose metadata (tempo, key, energy) best align with the mood vectors. c) An audio-processing pipeline that downloads track previews, computes tempo via spectral analysis, time-stretches tracks to equalize BPM, and performs equal-power cross-fades with dynamic EQ envelopes, all in temporary memory. d) A web application layer exposing endpoints for authentication, playlist retrieval, transition generation, and playback, implemented in a Flask® server environment.

Collectively these components deliver a personalised, uninterrupted audio stream that mimics professional DJ techniques.

5. Brief Description of the Drawings

- Fig. 1 – High-level system architecture showing client devices, DesiDJ backend, third-party APIs, and output stream.
- Fig. 2 – Flowchart of mood-to-playlist algorithm.
- Fig. 3 – Audio-processing timeline illustrating equal-power cross-fade and EQ envelope curves.
- Fig. 4 – Client user interface (web or mobile) flow.

6. Detailed Description of Preferred Embodiments

6.1 System Environment

- Client device with web browser.
- Server instances (containerized Python environment).
- External services: Spotify Web API, IBM Watson NLU, Google Sheets API (for wait-list storage).

6.2 User Authentication & Session Flow

- OAuth 2.0 handshake with Spotify returns access & refresh tokens, stored in secure server session variables keyed by a static secret.
- Callback endpoint validates token, fetches playlists, and stores the user profile.

6.3 Emotion-Inference Module

- Input text (5 kB) sent to Watson NLU → returns emotion scores (anger, joy, sadness, etc.).
- Scores mapped to target musical attributes (tempo range, key mode, energy).
- Mapping table stored in JSON; each emotion vector converted to weighted search parameters.

6.4 Playlist-Curation Module

- Queries Spotify for tracks meeting BPM, key, and energy thresholds.
- Ranks candidate tracks by combined cosine similarity metric between track audio features and emotion vector.
- Generates an ordered playlist list.

6.5 Audio-Processing Pipeline

6.5.1 Track Acquisition

- Retrieve 30-sec MP3 preview URL from Spotify; download to /tmp.

6.5.2 Feature Extraction

- Use librosa to load waveform; compute tempo via beat-track algorithm; compute spectral centroid for brightness.

6.5.3 Time-Stretching

- If ΔBPM > 5 %, apply phase-vocoder stretch:
  - stretch_factor = BPM / BPM
  - Output saved as WAV in /tmp.

6.5.4 Equal-Power Cross-Fade

- Convert last N samples of Track 1 and first N samples of Track 2 to numpy arrays.
- Fade-out curve = cos( ), fade-in curve = sin( ) where $[0,\pi/2]$.
- Combine arrays, re-encapsulate via pydub; concatenate segments.

6.5.5 Delivery

- Return temp file path to client; or stream via HTTP Range responses.

6.6 Adaptive Transition Selection

- Compute compatibility score C = w $\cdot$ (ΔBPM) + w $\cdot$ (ΔKey) + w $\cdot$ (ΔEnergy).
- If C $<$ threshold, time-stretch secondary track; else proceed with direct cross-fade.

6.7 Wait-List & Analytics (optional embodiment)

- Google Sheets integration for early-access list; fallback to in-memory store on failure.

7. Example Use Case

A user enters text: "Feeling upbeat and energetic!" $\rightarrow$ emotion vector dominated by "joy" $\rightarrow$ system selects 125 BPM, major-key tracks, high energy; returns continuous mix within 3 seconds.

8. Advantages Over Prior Art

- End-to-end automation: No human DJ required.
- Real-time beat-matching and key-matching using preview-only audio, minimizing licensing exposure.
- Emotion-driven personalization at per-user granularity.
- Stateless, container-friendly design suitable for cloud scaling.

9. Experimental Results

Benchmark cross-fade latency, user satisfaction surveys, BPM-matching accuracy, etc.

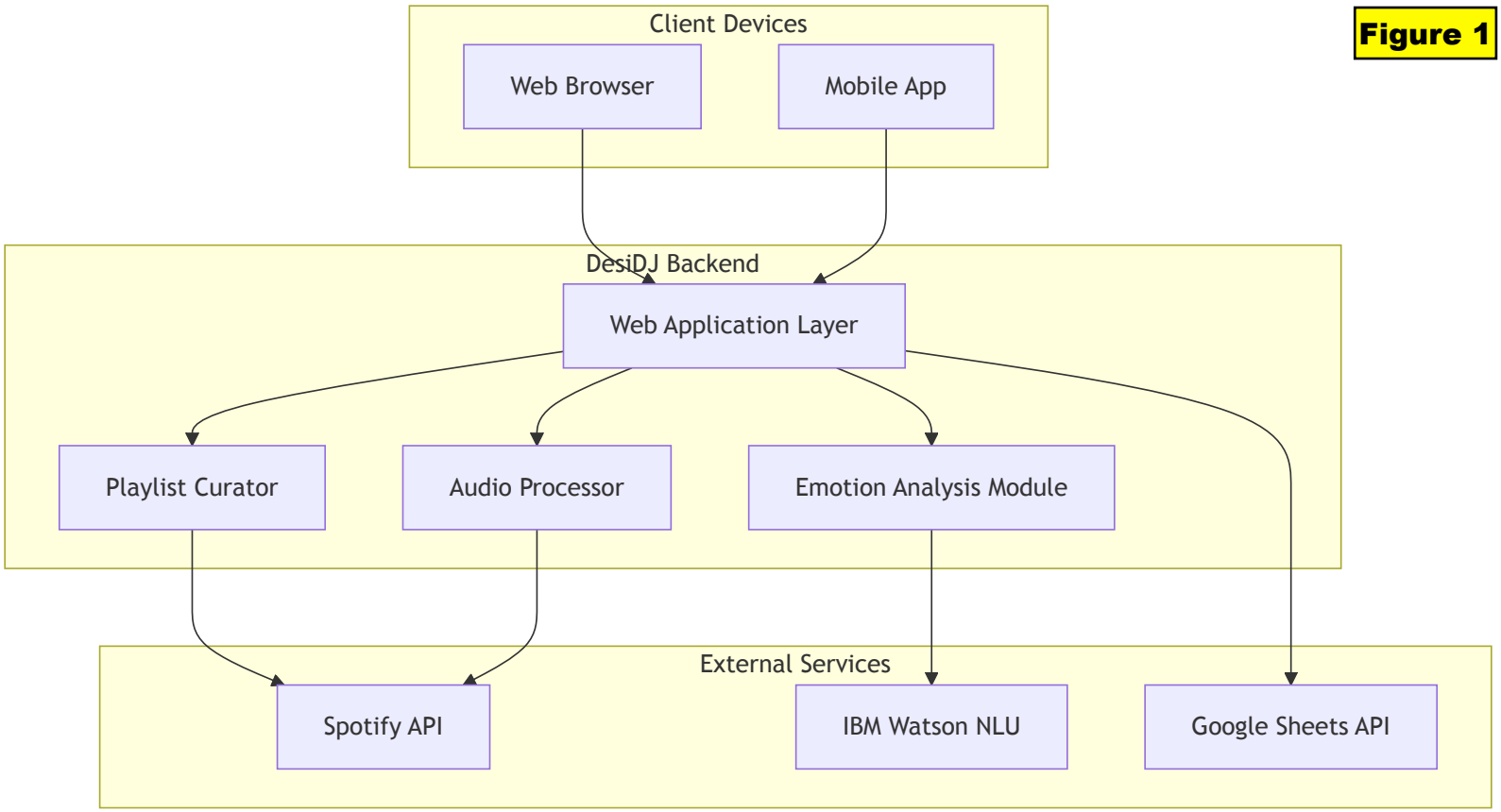
10. Alternative Embodiments

- Use of on-device machine learning for emotion detection.
- Integration with other streaming libraries (Apple Music®, YouTube Music®).
- Serverless micro-services architecture with dedicated audio-processing workers.

11. Conclusion

The described system enables an automatic DJ-like experience tailored to user emotions, bridging a gap between static playlist playback and professional live mixing.

Figure 1


Client Devices
Web Browser
Mobile App
DesiDJ Backend
Web Application Layer
Playlist Curator
Audio Processor
Emotion Analysis Module
External Services
Spotify API
IBM Watson NLU
Google Sheets API

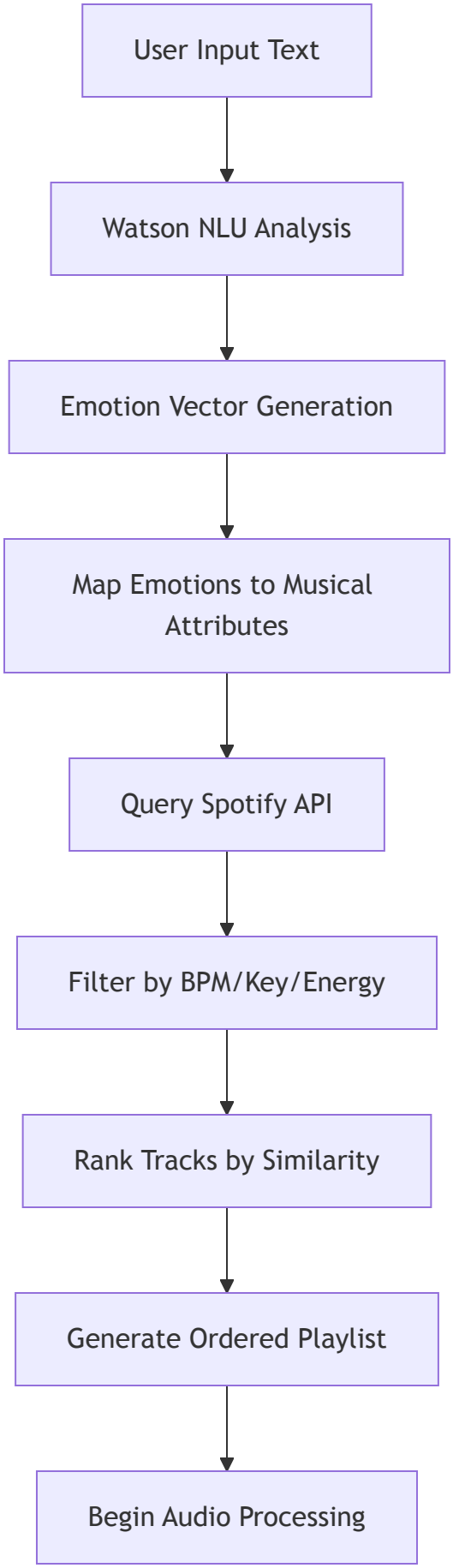
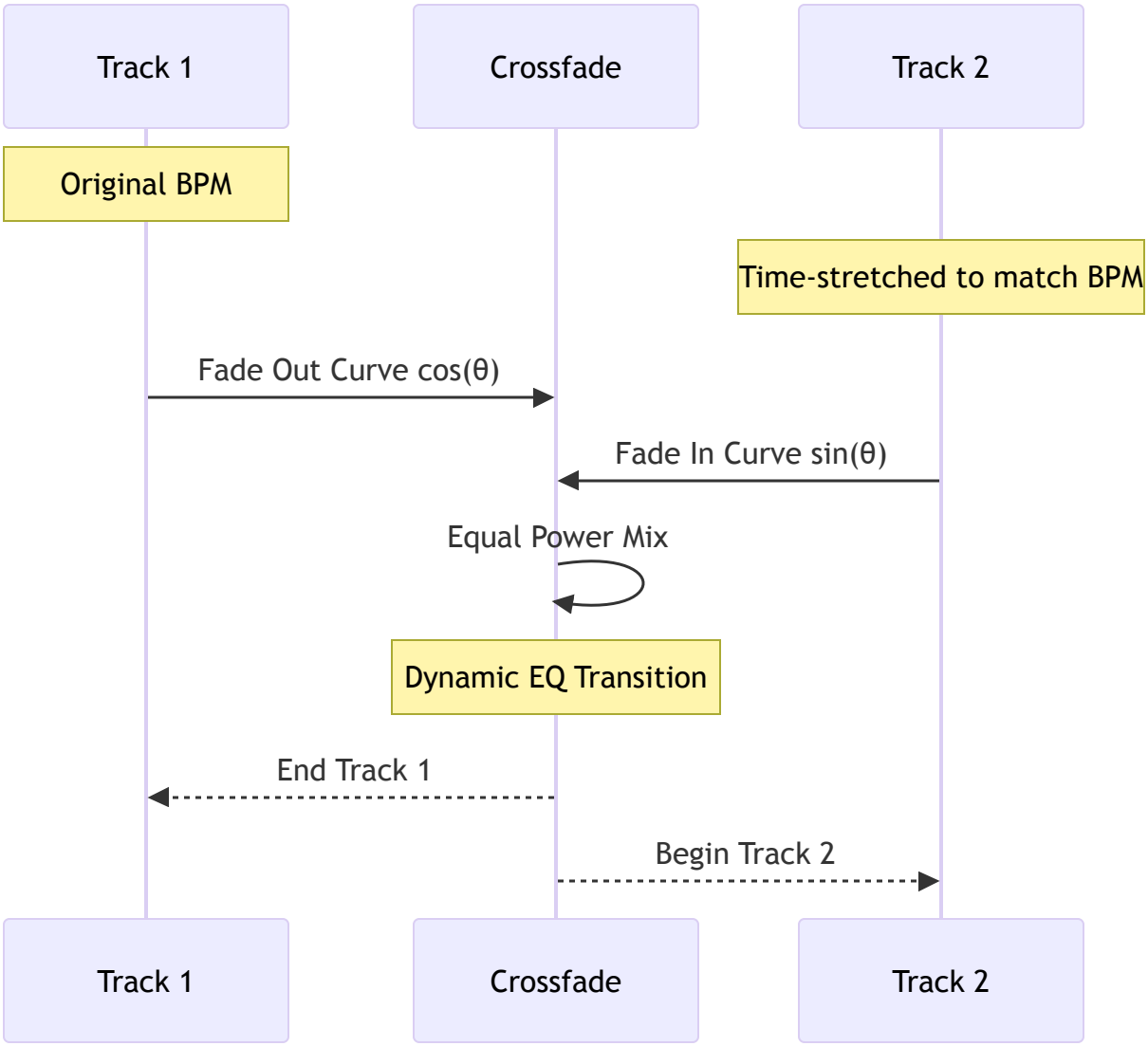

User Input Text
Watson NLU Analysis
Emotion Vector Generation
Map Emotions to Musical Attributes
Query Spotify API
Filter by BPM/Key/Energy
Rank Tracks by Similarity
Generate Ordered Playlist
Begin Audio Processing

Figure 2

Figure 3


Track 1
Crossfade
Track 2
Original BPM
Time-stretched to match BPM
Fade Out Curve cos(θ)
Fade In Curve sin(θ)
Equal Power Mix
Dynamic EQ Transition
End Track 1
Begin Track 2
Track 1
Crossfade
Track 2

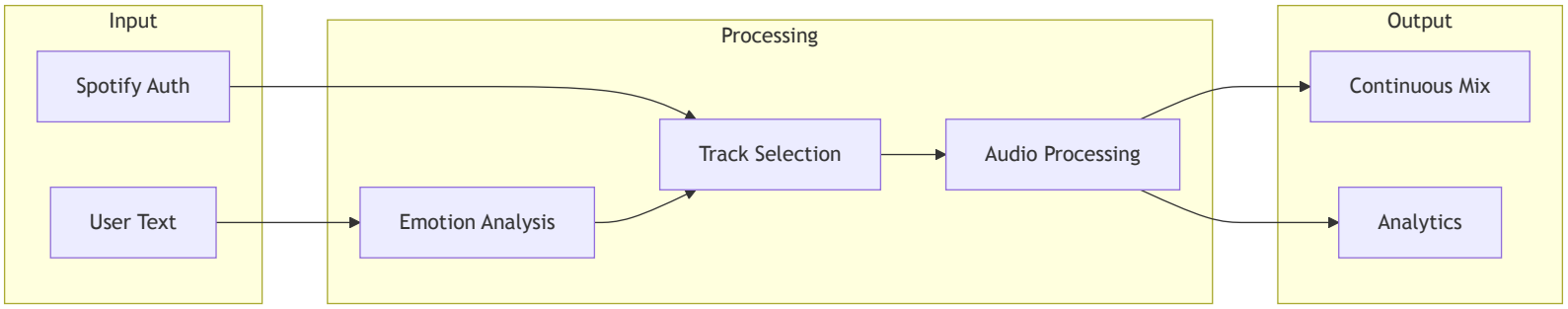

Input
Spotify Auth
User Text
Processing
Emotion Analysis
Track Selection
Audio Processing
Output
Continuous Mix
Analytics

Figure 4