AudioMuse-AI: Smart Playlist Generation for Your Jellyfin Homelab

NeptuneHub

June 9, 2025

What is AudioMuse-AI?

AudioMuse-AI is a Dockerized application that intelligently transforms your Jellyfin music library. It uses deep audio analysis and AI to understand your music's characteristics, generating smart, curated playlists. Think of it as an intelligent DJ for your homelab, enhancing your self-hosted music experience.

Why AudioMuse-AI is Useful for Your Homelab?

- **Effortless Playlists:** Say goodbye to manual curation; AudioMuse-AI automates playlist creation for you.
- **Rediscover Music:** Uncover hidden gems by grouping tracks based on their unique audio features.
- **Mood-Based Listening:** Generate playlists perfectly suited for any mood or activity, from high-energy workouts to relaxing evenings.
- **Seamless Jellyfin Integration:** Playlists are created directly within your Jellyfin library, ready to play.
- **Open Source & Customizable:** You have full control, allowing you to fine-tune analysis and clustering parameters to your liking.

Quick Start: Deploying in Your Homelab

AudioMuse-AI is designed for easy deployment using Docker Compose (for local setups) or Kubernetes (K3S).

- **Prerequisites:** Ensure you have Docker and Docker Compose (local) or a K3S cluster with `kubectl` (Kubernetes).
- **Configuration:** Update provided `docker-compose.yaml` or `deployment.yaml` files. Key settings include your Jellyfin URL, API token, user ID, and database/Redis credentials. You can also configure the Gemini API key for AI naming.
- **Deployment:**
  - For Docker Compose: `docker compose up -d --scale audiomuse-ai-worker=2`.
  - For Kubernetes: `kubectl apply -f deployment.yaml`.
- **Access:** Once deployed, access the web UI at `http://<YOUR-IP>:8000`.

How It Works: Workflow Overview

AudioMuse-AI transforms your music through a structured workflow:

1. **User Initiation:** You start analysis or clustering tasks from the web UI.
2. **Task Queuing:** Jobs are sent to a background queue (Redis Queue) for asynchronous processing, keeping the UI responsive.
3. **Parallel Processing:** Multiple workers handle tasks simultaneously, speeding up analysis for large libraries.
4. **Analysis Phase:**
   - Audio is downloaded from Jellyfin and analyzed by **Essentia** and **TensorFlow**.
   - Features like tempo, key, energy, moods, and genres are extracted.
   - Results are saved to a PostgreSQL database.
5. **Clustering Phase:**
   - An advanced evolutionary algorithm groups analyzed tracks into cohesive playlists.
   - Optionally, AI models (Ollama or Gemini) generate creative playlist names.
   - Final playlists are created directly in Jellyfin.

Deep Dive: Audio Analysis Algorithm

AudioMuse-AI extracts rich features from your music library:

- **Audio Loading & Preprocessing:** Tracks are downloaded from Jellyfin and consistently resampled to 16000 Hz.
- **Core Feature Extraction (Essentia):**
  - **Tempo:** Uses `RhythmExtractor2013` for BPM.
  - **Key & Scale:** Identifies musical key and scale.
  - **Energy:** Calculates normalized average energy for intensity.
- **Embedding Generation (TensorFlow & Essentia):**
  - **MusiCNN Embeddings:** A 200-dimensional vector captures high-level semantic information using pre-trained TensorFlow models.
- **Prediction Models (TensorFlow & Essentia):**
  - Predicts primary tags/genres (e.g., 'rock', 'pop').
  - Predicts specific moods and characteristics like 'danceable', 'happy', 'sad'.
- **Feature Vector Preparation:** All extracted features are normalized and standardized into a single numerical vector per track, ready for clustering.

Example: Audio Analysis Output

Analyzed Track: "My Awesome Song by Great Artist"

After analysis, a track yields detailed attributes and a numerical feature vector:

- **Metadata:**
  - **Tempo:** 120.5 BPM
  - **Key:** D Major
  - **Energy:** 0.72 (normalized)
- **Moods & Other Features (Predicted Scores):**
  - **Moods:** {'happy': 0.88, 'energetic': 0.75, 'party': 0.62, ...}
  - **Other Features:** {'danceable': 0.91, 'aggressive': 0.15, ...}
- **Resulting Feature Vector (Simplified):**

$$[\underbrace{0.65}_{\text{norm. tempo}}, \underbrace{0.82}_{\text{norm. energy}}, \underbrace{0.88}_{\text{happy}}, \underbrace{0.65}_{\text{rock}}, \underbrace{0.91}_{\text{danceable}}, \underbrace{0.15}_{\text{sad}}, \ldots]$$

This comprehensive vector forms the basis for subsequent clustering.

Deep Dive: Clustering & Evolutionary Approach

AudioMuse-AI intelligently groups songs into playlists using various clustering algorithms, such as K-Means, DBSCAN, and Gaussian Mixture Models (GMM). The key to finding optimal playlist configurations is an advanced **Monte Carlo evolutionary approach**.

- **Multiple Iterations:** The system runs clustering hundreds or thousands of times (default 1000), experimenting with different parameters for the chosen algorithms and PCA.
- **Evolutionary Strategy:** It "learns" from good solutions by remembering "elite" parameter sets. These elite sets are then mutated to explore their "neighborhood" and find even better configurations.
- **Comprehensive Scoring:** Each clustering outcome is evaluated with a composite score. This score balances factors like overall playlist diversity, the consistency (purity) within individual playlists, and internal clustering metrics.
- The configuration that yields the best overall score is then chosen to generate your final Jellyfin playlists.

Example: Clustering Output (without AI Naming)

Playlists Generated by Clustering Algorithm (Default Naming)

After the clustering algorithm groups similar songs, playlists are named based on their predominant characteristics extracted from the cluster centroids:

- **Playlist 1:** `Rock_Fast_Danceable`
  - Songs in this playlist exhibit high scores for Rock genre, Fast tempo, and Danceable attributes.
- **Playlist 2:** `Chillout_Medium_Relaxed`
  - This playlist contains songs characterized by Chillout moods, Medium tempo, and a Relaxed vibe.
- **Playlist 3:** `Pop_Upbeat_Happy`
  - A collection of songs identified with Pop characteristics, an Upbeat tempo, and a Happy mood.

These names are descriptive but can sometimes lack a creative touch.

AI Playlist Naming: Adding Creativity

Add a creative touch to your generated playlists!

- After clustering, AudioMuse-AI can use an AI model (Ollama or Google Gemini) to generate descriptive, human-readable names.
- **Input to AI:** Key characteristics from the cluster's centroid (moods, tempo, etc.) and sample songs are provided.
- **Prompt Engineering:** Carefully crafted prompts guide the AI to generate concise names reflecting the playlist's vibe.

Example: AI Playlist Naming

AI-Generated Playlist Names

Consider a cluster identified with high "Rock" mood, "Fast" tempo, and "Danceable" characteristics.

- **Without AI (Default Name):**

  `Rock_Fast_Danceable`

  This name is functional but purely descriptive.

- **With AI (Evocative Name):** The AI processes the cluster's attributes and suggests a more engaging name. For the same cluster, it might propose:

  Rock N' Roll Rumble

  This name instantly conveys energy and style. Other possibilities could include: *High-Energy Rock Mix*, *Amped Up Anthems*, or *Dancefloor Rockers*.

This feature transforms generic labels into appealing and memorable playlist titles, enhancing discoverability.

Key Technologies Under the Hood

Built with a robust stack of open-source technologies:

- **Flask:** Lightweight web interface and API.
- **Redis Queue (RQ):** Background job processing for analysis and clustering.
- **Essentia-TensorFlow:** Core audio analysis, feature extraction, and deep learning models.
- **scikit-learn:** Machine learning algorithms for clustering and dimensionality reduction.
- **PostgreSQL:** Persistent storage for analyzed track data and playlist structures.
- **Ollama / Google Gemini API:** Optional AI models for intelligent playlist naming.
- **Jellyfin API:** Direct integration to manage your media and playlists.
- **Docker / OCI Containers:** Ensures easy and consistent deployment in your homelab.

Future Possibilities

This is just the beginning!

- **Integration into Music Clients:** Directly use for playlist creation or instant mixes within your favorite players.
- **Jellyfin Plugin:** Seamless integration directly within the Jellyfin UI for a single, easy-to-use front-end.
- **Cross-Platform Sync:** Export playlists to .m3u or sync to external platforms.

Thank You!

AudioMuse-AI: Bring Smart Playlists to Your Jellyfin Homelab

`https://github.com/NeptuneHub/AudioMuse-AI`