🔹 Overview

Our web application is designed to provide AI-driven music recommendations based on uploaded images and user mood analysis. The platform also creates personalized playlists based on user activity and allows for collaborative group folders where users can add songs.

Key Features:
✔ Upload images & get AI-based music recommendations
✔ AI-powered personalized playlists based on user mood & activity
✔ Group Folders (Joint Albums) where users collaborate on music collections
✔ Secure authentication with Auth0
✔ IPFS storage for images using Pinata
✔ Database management with MongoDB
✔ AI-driven recommendations with Google Gemini AI

🛠️ Technologies & Tools Used

Technology	Usage in the Project
Pinata (IPFS Storage)	Stores uploaded images on **IPFS** for **decentralized & permanent storage**
MongoDB	Stores user data, posts along with the credentials to show what users it belongs to, song recommendations, posts' comments, and **group folders**
Auth0	Manages **user authentication & authorization** securely
Google Gemini AI	**Analyzes images & mood**, **recommends songs**, and **personalizes playlists**
Next.js	Handles **frontend & backend API routes** efficiently
Spotify API	Fetches **song metadata & streaming links and the 30 seconds songs previews to include under the picture.**
React (Frontend)	Creates an **interactive UI** for the app

🔹 How We Use These Tools in Our Project

📌 1. Pinata (IPFS Storage)

- Used to **store uploaded images** permanently on IPFS.
- Each uploaded image generates a **unique IPFS hash (CID)** that allows users to **retrieve their images anytime**.

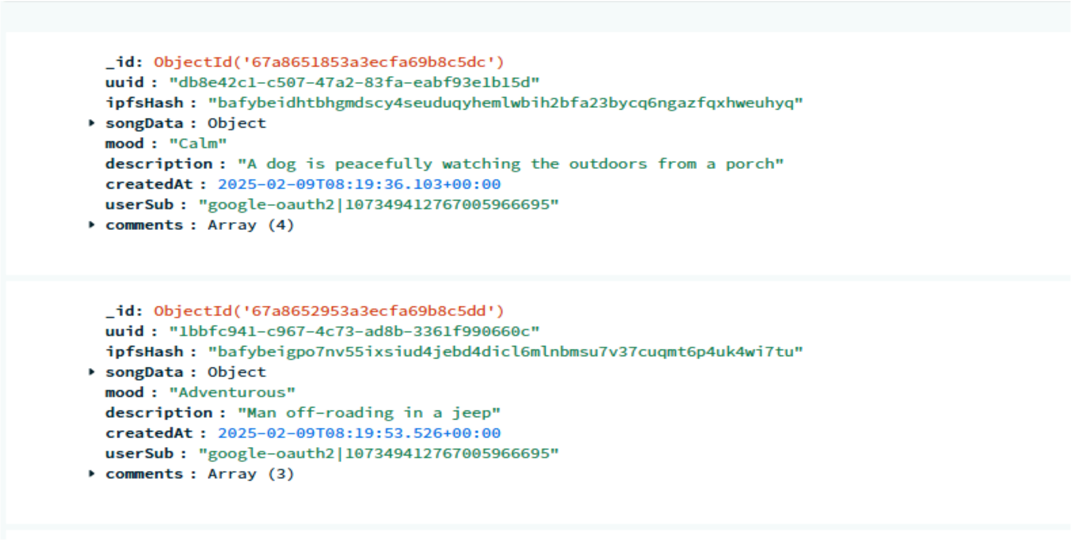
Example Workflow:
1️⃣User uploads an image.
2️⃣Image is sent to **Pinata via API** → **stored on IPFS**.
3️⃣Pinata **returns an IPFS CID (hash)**.
4️⃣CID is **stored in MongoDB** & linked to the user's profile.

Why IPFS?
✔ **Decentralized storage** (no risk of deletion)
✔ **Fast retrieval & security**
✔ **Permanently accessible links**

📌 2. MongoDB (Database for Users, Images, Songs, & Folders)

- Stores **user accounts, uploaded images, song recommendations, and group folders**.
- Every image upload **links to a user ID** and **contains AI-analyzed metadata**.
- **Group Folders** allow users to **collaborate on song collections**.

Database Structure Example (`uploads` Collection):



```
  _id: ObjectId('67a8651853a3ecfa69b8c5dc')
  uuid : "db8e42c1-c507-47a2-83fa-eabf93e1b15d"
  ipfsHash : "bafybeidhtbhgmdscy4seuduqyhemlwbih2bfa23bycq6ngazfqxhweuhyq"
▸ songData : Object
  mood : "Calm"
  description : "A dog is peacefully watching the outdoors from a porch"
  createdAt : 2025-02-09T08:19:36.103+00:00
  userSub : "google-oauth2|107349412767005966695"
▸ comments : Array (4)

  _id: ObjectId('67a8652953a3ecfa69b8c5dd')
  uuid : "1bbfc941-c967-4c73-ad8b-3361f990660c"
  ipfsHash : "bafybeigpo7nv55ixsiud4jebd4dicl6mlnbmsu7v37cuqmt6p4uk4wi7tu"
▸ songData : Object
  mood : "Adventurous"
  description : "Man off-roading in a jeep"
  createdAt : 2025-02-09T08:19:53.526+00:00
  userSub : "google-oauth2|107349412767005966695"
▸ comments : Array (3)
```

✔ **Efficient & scalable** NoSQL database
✔ **Stores structured metadata for AI recommendations**
✔ **Ensures quick retrieval of user activity**

📌 3. Auth0 (User Authentication & Authorization)

- Manages **secure login & user sessions**.
- Ensures **only authenticated users** can upload images, access recommendations, and manage group folders.

How It Works:
1️⃣User logs in via **Google/Auth0 credentials**.
2️⃣Auth0 **issues an access token**.
3️⃣The token is **used for all protected API requests**.
4️⃣Users can **only delete/edit their own uploads**.

Benefits:
✔ **OAuth & Social Login Support**
✔ **JWT-Based Authentication for Secure APIs**
✔ **Prevents unauthorized access**

📌 4. Google Gemini AI (Image Analysis & Music Recommendations)

- Analyzes **uploaded images** to detect **objects, colors, and emotions**.
- Uses **sentiment analysis** to determine **the user's mood**.
- Fetches **music recommendations based on AI mood analysis**.
- Learns from **user activity to create dynamic, personalized playlists**.

Example Workflow:
1️⃣User uploads a **selfie with a happy expression**.
2️⃣**Gemini AI detects "Happy Mood"** → recommends **Upbeat Pop Songs**.
3️⃣AI stores user preferences in MongoDB & **adapts over time**.
4️⃣The more the user **interacts**, the better the AI understands **their music taste**.

✔ **Personalized listening experience**
✔ **Improves recommendations over time**
✔ **Dynamic mood-based playlist generation**

🔹 Features & Functionalities

📌 1. Upload Images & Get AI-Powered Song Recommendations

- Users **upload images** that represent their **mood or moment**.
- **Gemini AI analyzes** the image for **mood detection**.
- AI **suggests songs that match the mood**.

Example:
📸 **User Uploads a Sunset Photo** → AI detects **Calm Mood** → Suggests **Lo-Fi or Jazz Music**.

📌 2. AI-Powered Personalized Playlists

- Based on **user listening history & uploaded moods**.
- AI **tracks engagement with songs** to refine recommendations.
- Playlists are **updated dynamically** as the user interacts.

Example:
🎧 **User frequently uploads "Workout" images** → AI **creates an Energizing Gym Playlist**.

📌 3. Group Folders (Joint Albums)

- Users can **create shared folders** to **collaborate on music collections**.
- A song can **only belong to one folder** at a time.
- Members **can add & remove songs** but **cannot duplicate them across folders**.

Example Workflow:
1️⃣ **User A creates a "Road Trip Vibes" folder**.
2️⃣ **User B adds songs** to the shared folder.
3️⃣ The playlist updates **for all members in real-time**.

✔ **Encourages music collaboration**
✔ **Prevents duplicate song entries**
✔ **Creates a shared music experience**
