Music Playlist Manager – Project Proposal

1. Introduction

Digital music consumption is one of the most popular activities in today's world, cutting across age, profession, and location. Whether you're a student who studies with music, a DJ creating setlists, or someone simply curating their favorite tracks for morning runs, managing your music efficiently matters. Unfortunately, most music library tools either oversimplify the experience or overcomplicate it with features the average user doesn't need.

This project proposes a solution to that gap. The **Music Playlist Manager** is a clean, intuitive, and feature-rich web application that lets users upload their music files, create and customize playlists, search tracks easily, and track their listening habits. It is aimed at users who prefer a personalized and independent music management platform, rather than relying solely on mainstream streaming services.

The application will also support playlist sharing, encouraging collaboration, discovery, and a stronger user community. This is not a streaming service; it's a personal music library system that gives the user full control.

2. Problem Statement

Managing digital music libraries can be surprisingly difficult, especially when they grow large or span multiple folders and devices. Despite the popularity of music, few applications offer a simple yet powerful interface that balances user-friendliness with useful functionality.

Here are the key problems this project addresses:

a. Disorganization in Music Collections

Users often store their songs in local folders, USB drives, cloud storage, or on various devices, with no centralized management system. This leads to duplicates, missing files, or general confusion.

b. Limited Playlist Tools

Many existing music players offer basic playlists with little customization. There's often no way to create dynamic playlists based on genre, mood, or listening history.

c. Lack of Personal Storage

Mainstream apps don’t let you upload and organize your own MP3s or audio files. There’s a need for a personal space where users control what music is added and how it’s categorized.

d. Weak Discovery and Search

When a collection grows, finding the right song becomes difficult. Most apps only offer simple title or artist search, lacking advanced filters or sorting options.

e. No Listening History

A user might want to replay a song they listened to yesterday but can’t remember the name. Without a recently played section, they are left guessing or searching manually.

f. Poor User Interfaces

Some tools feel outdated, are not mobile-friendly, or overwhelm users with cluttered menus and options they don’t need.

In short, users want a tool that gives them freedom, organization, and a smooth experience—something that feels personal and easy to use.

3. Proposed Solution

Our proposed solution is to build a responsive, full-stack **Music Playlist Manager** web application that allows users to manage their music library entirely online.

Key Features

- **User Authentication**
  Users can sign up, log in, and manage their own accounts securely. Passwords will be encrypted and sessions managed safely using industry-standard tools.

- **Music File Upload & Storage**
  Users can upload their own MP3 or audio files. The system will store these files securely, allowing easy access and playback from anywhere.

- **Playlist Creation & Editing**
  Users can create multiple playlists, add or remove songs, rearrange the order, rename them, or delete them. Playlists can be themed by mood, activity, or event.

- **Search and Filter System**
  Users can search by artist, song title, genre, or even date added. Filters allow users to

quickly narrow down their collection.

- **Listening History**
  A “Recently Played” section allows users to track their listening habits and rediscover songs.

- **Playlist Sharing (Public/Private)**
  Users can choose to keep playlists private or make them public. Public playlists can be shared through a link and accessed by others.

- **Modern, Responsive User Interface**
  Built with mobile-first design principles so the app works well on phones, tablets, and computers.

Technical Design

The application will be built using **object-oriented programming** to ensure the codebase is modular, reusable, and easy to maintain. The backend and frontend will communicate through a **RESTful API**, which separates logic and improves scalability.

For fast processing and memory management, **Redis** will be used to store recent activity such as recently played songs. All audio files will be stored securely, with validation to avoid unsupported or corrupted uploads.

4. Technologies to Be Used

This project combines modern technologies across the full stack, allowing for both speed and flexibility.

Backend Technologies

- **Node.js**: JavaScript runtime used for server-side scripting

- **Express.js**: Web framework for building API routes

- **MongoDB + Mongoose**: NoSQL database and ORM for storing users, music files, playlists, and activity logs

- **Redis**: For caching and managing listening history

- **JWT (JSON Web Tokens)**: For secure authentication tokens
- **Multer**: For handling audio file uploads
- **Joi**: Validates user inputs to prevent bad data
- **bcryptjs**: Encrypts user passwords for secure storage

Frontend Technologies

- **HTML5 & CSS3**: Markup and styling for the app
- **Vanilla JavaScript**: Provides interactivity on the client side
- **Responsive Design Principles**: Ensures accessibility across screen sizes and devices

Development Tools

- **Git & GitHub**: Version control and team collaboration
- **Jest**: Automated unit testing
- **Supertest**: API endpoint testing
- **Nodemon**: Monitors backend code changes during development

---

5. Project Timeline and Milestones

Day	Milestone	Activities
Day 1	Project Setup & Planning	Create repo, design database schema, assign roles, define workflow
Day 2	Backend Development	Build authentication, API endpoints, file uploads, database models
Day 3	Frontend Development	Build login pages, dashboard UI, playlist interface, embed audio player

Day 4	Integration	Connect frontend and backend, implement Redis, add search and sharing
Day 5	Testing & Documentation	Write unit tests, finalize API documentation, fix UI issues
Day 6	Final Review & Deployment	Fix bugs, polish UI, prepare demo, and deploy the application

6. Expected Outcomes

At the end of this project, we expect to deliver a complete and polished web application that includes:

- A secure and well-tested user management system
- A fast, intuitive, and mobile-friendly user interface
- Playlist creation and editing features that meet everyday needs
- Smart search and recently played tracking using Redis
- Full technical documentation (API, setup, and user guide)
- Clean and modular codebase using OOP principles
- A deployment-ready version of the app
- Presentation materials showcasing functionality and architecture

7. Conclusion

The Music Playlist Manager is more than a coding project—it’s a real solution to a real problem faced by everyday users who value control, simplicity, and personalization in their digital music experience.

This project not only showcases our team's skills in full-stack development, software design, testing, and collaboration, but also delivers a product that can be further developed or adapted into a larger music organization platform. By focusing on real-world problems and implementing practical solutions, this project represents what effective software engineering is all about.