Project Documentation

Rhythmic Tunes: Your Melodic Companion

1. Introduction

Project Title: Rhythmic Tunes: Your Smart Retail Companion

Team ID: NM2025TMID39970

Team Leader: YOSHINE L [sac2427csc5409@ssacollegechennai.com]

Team Members:

– PRIYANKA G [sac2427csc5435@ssacollegechennai.com]

– POOJA P [sac2427csc5768@ssacollegechennai.com]

– REVATHY V [sac2427csc5323@ssacollegechennai.com]

–PUSHPA BIBIYA J [sac2427csc5296@ssacollegechennai.com]

Backend: Node.js + Express.js for APIs and server logic

Database: MongoDB (stores users, songs, playlists, ratings, favorites)

2) Project Overview

Purpose:

Rhythmic Tunes is a web application that allows users to explore, upload, and manage songs and playlists. It provides personalized collections, search filters, and a seamless listening experience.

Features:

– Upload and browse songs with cover images

– Categorization by genre, artist, and duration

– Playlist creation and management

– User accounts with favorites and listening history

– Admin dashboard for content moderation

3) Architecture

Frontend: React.js with Material-UI for a visually appealing interface

Backend: Node.js + Express.js for APIs and server logic

Database: MongoDB (stores users, songs, playlists, ratings, favorites)

4) Setup Instructions

Prerequisites:

– Node.js

– MongoDB

– Git

– Visual Studio Code

Installation Steps:

Git clone

# Install client dependencies

Cd client

Npm install

# Install server dependencies

Cd ../server

Npm install

5) Folder Structure

Rhythmic-tunes/

├── client/ # React frontend

│ ├── components/

│ ├── pages/

│ └── assets/

├── server/ #Node.js backend

│ ├── routes/

│ ├── models/

│ ├── controllers/

│ └── middleware/

└── README.md

6) Running the Application

Frontend: Cd client npm start

Backend: Cd server npm start

AcManual testing for song upload, search, and playlist creation

Tools: Postman, Chrome DevTools, Jest for unit tests

7) API Documentation

User:

– POST /api/user/register – Create account

– POST /api/user/login – Log in

Songs:

– POST /api/songs/upload – Upload song

– GET /api/songs/:id – Get song details

Playlists:

– POST /api/playlists/create

– GET /api/playlists/:userId

8) Authentication

– JSON Web Token (JWT) for login sessions

– Middleware to protect private routes

9) User Interface

– Landing Page with featured songs

– Song Page with details and play option

– Playlist Dashboard

– Admin Panel for content moderation

10) Testing

Manual testing for song upload, search, and playlist creation

Tools: Postman, Chrome DevTools, Jest for unit tests

11) Known Issues

– Audio upload may be slow on some networks

– Limited offline playback

– Occasional delay in song loading on slow networks

– Volume slider not persisting between sessions

12) Future Enhancements

– Recommendation engine using AI

– Collaborative playlists

– Mobile app version

cess: Visit http://localhost:5173

