# Project Documentation

## Rhythmic Tunes: Your Melodic Companion

## 1. Introduction

• Project Title: Rhythmic Tunes: Your Melodic Companion

• Team ID: NM2025TMID45753

• Team Leader: SUMAIYA TAJ A (Mail id: [sumaiyataj872@gmail.com](mailto:sumaiyataj872@gmail.com))

• Team Members:

– SUMITHRA A (Mail id: asumithra429@gmail.com )

– SUSMITHA M (Mail id: susmithasusmitha3396@gmail.com)

–  SWATHI R (Mail id: swathi3278705@gmail.com )

## 2. Project Overview

• Purpose: Rhythmic Tunes is designed as your melodic companion, enabling users to enjoy personalized, rhythmic, and tuneful experiences through a seamless interface.

• Features:

– Interactive and personalized music experience

– JSON server integration for data handling

– Modern frontend with Vite + React.js

– Local server execution and development ease

## 3. Architecture

• Frontend: React.js with Vite

• Backend: Node.js and Express.js for server logic

• Database: JSON Server simulating database functionality

## 4. Setup Instructions

• Prerequisites:

– Node.js (Download from https://nodejs.org/en/download/)

– Visual Studio Code

– Git (optional)

• Installation Steps:

# Download the project zip from SmartInternz portal and extract it

# Open the folder in VS Code

# Install dependencies using: npm install

# Start the frontend using: npm run dev

# Setup JSON server by navigating to db folder and running: json-server --watch db.json --port 3000

## 5. Folder Structure

Rhythmic-Tunes/

|-- client/ # React frontend

|\_\_ components/

|\_\_ pages/

|-- db/ # JSON server files

|\_\_ db.json

## 6. Running the Application

• Frontend:

cd client

npm run dev

• Backend (JSON Server):

cd db

json-server --watch db.json --port 3000

• Access: Visit http://localhost:5173 for frontend and http://localhost:3000 for JSON server

## 7. API Documentation

• Songs:

– GET /songs

– POST /songs

– PUT /songs/:id

– DELETE /songs/:id

## 8. Authentication

• JSON server handles simple data but does not include advanced authentication by default.

• Can be extended to JWT authentication if required.

## 9. User Interface

• Landing Page with music categories

• Player interface for songs

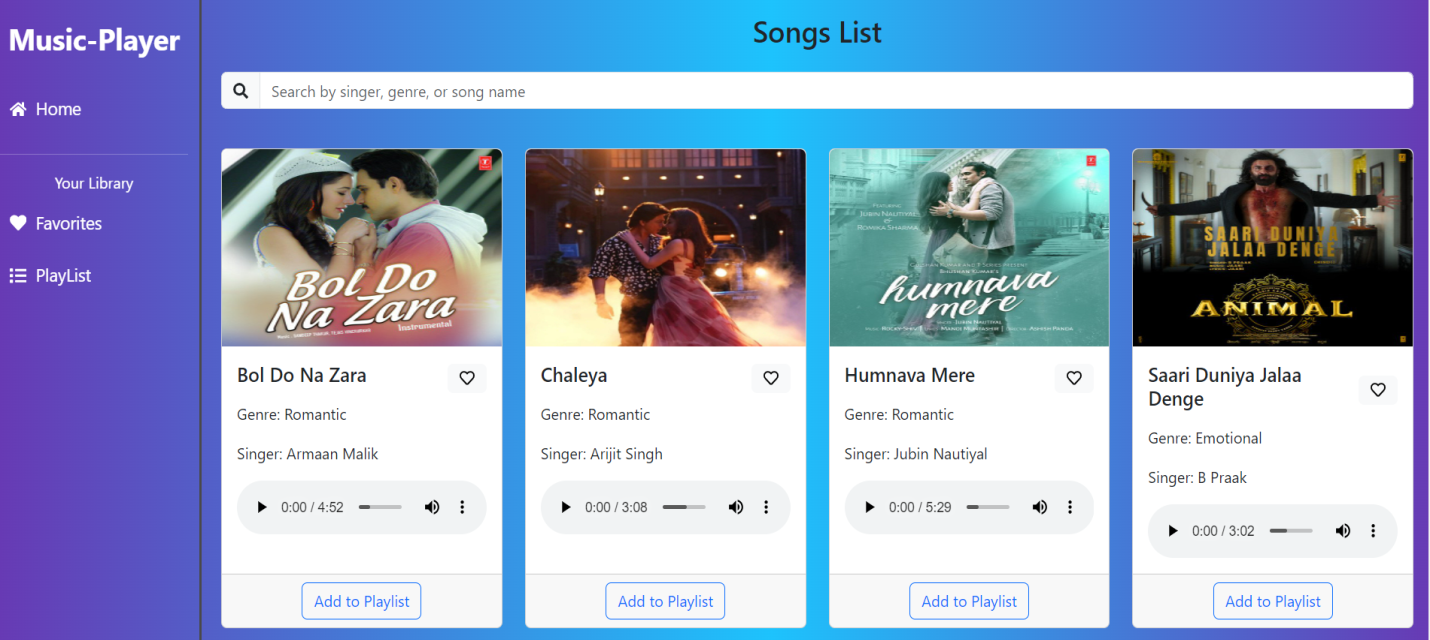
• Interactive controls for playback

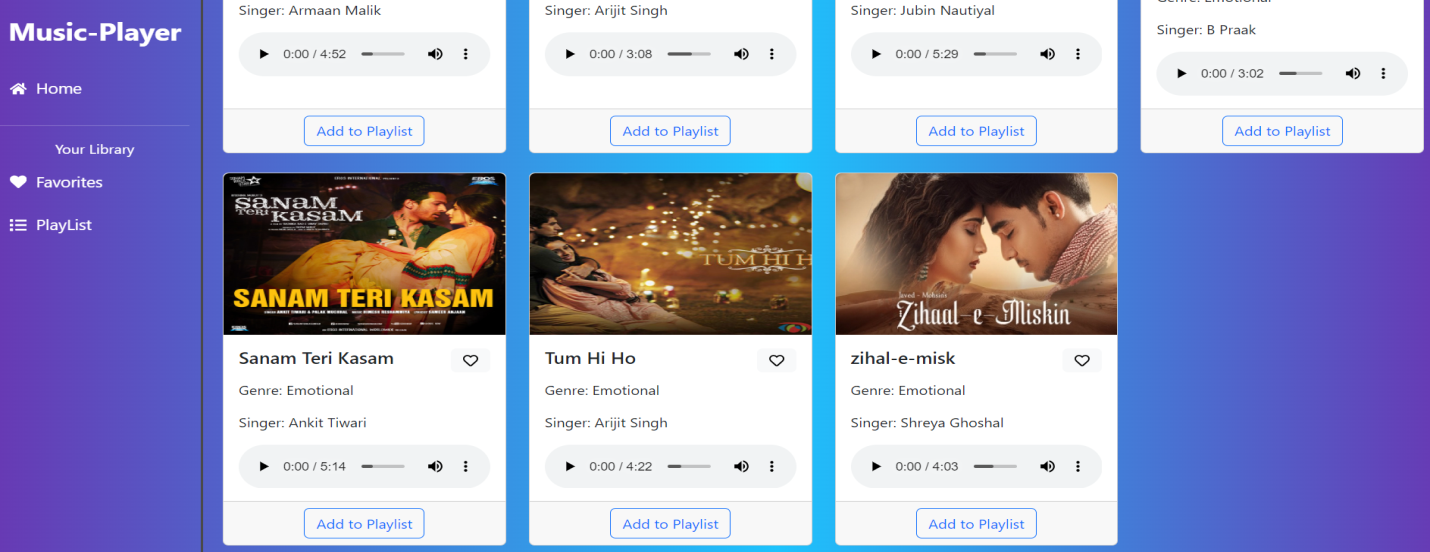
## 10. Testing

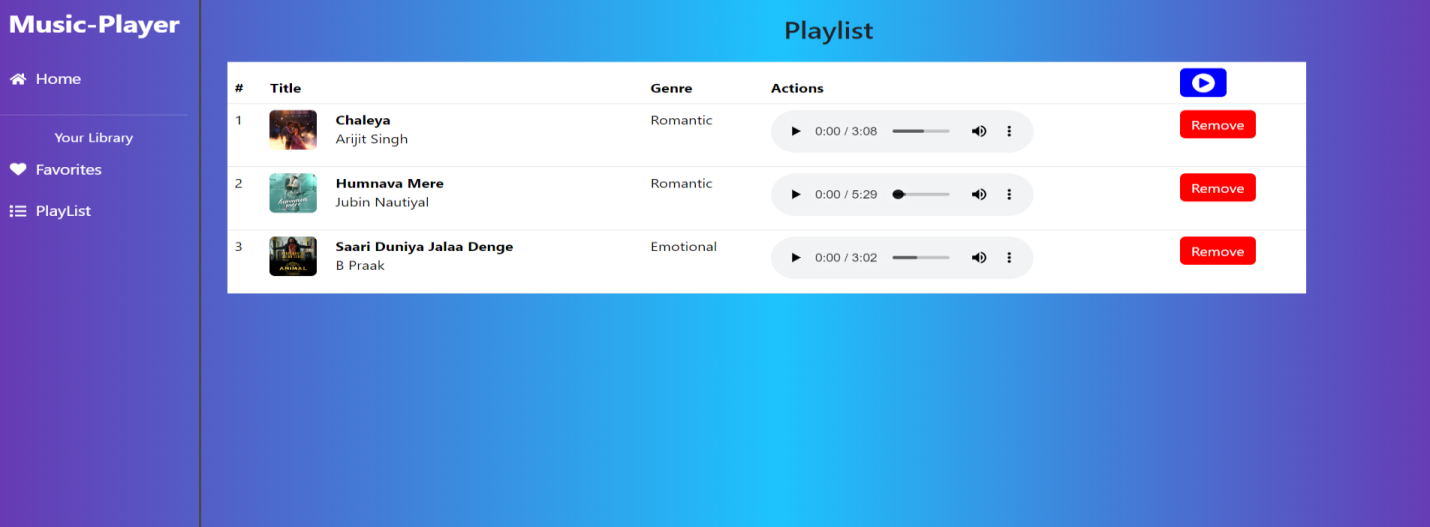
• Manual testing of features during execution.

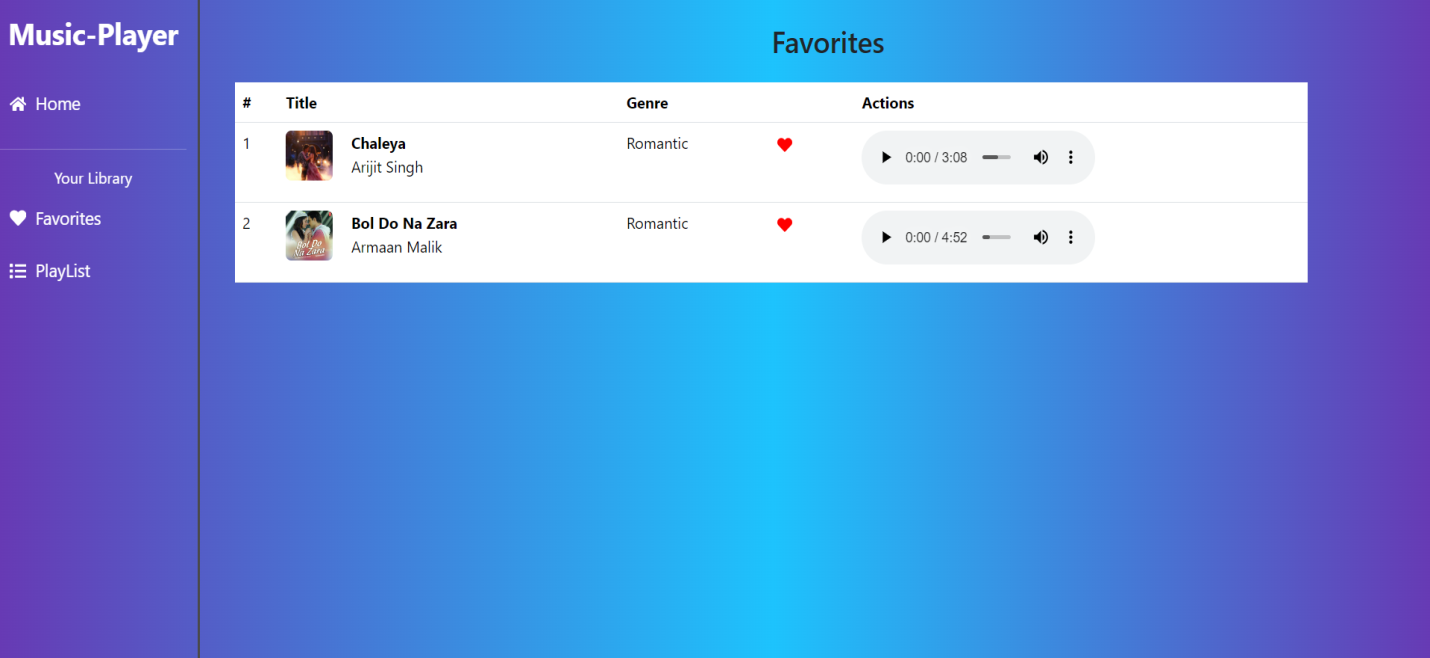
• Tools: Browser Developer Tools, Postman (for API endpoints).

## 11. Screenshots or Demo









## 12. Known Issues

• Limited to local environment execution without cloud deployment.

• JSON server is not a production-grade database.

## 13. Future Enhancements

• Add authentication and user profiles.

• Deploy the project on cloud platforms.

• Enhance music recommendation features with AI.