Project Documentation: Music Player (React + Vite)

# Introduction

Project Title: **Rhythmic Tunes: Your Melodic Companion**

Team Leader : AKILA N

Team Members:

ANNAPOORNA B

ANU T

ARIFA BEGUM R

# Purpose

The Music Player Frontend delivers a simple, modern, and responsive interface for users to browse, play, and control music tracks. Built with React and Tailwind CSS, it prioritizes a clean UI, smooth user experience, and scalability for future backend/API integration.

# Features

* Playback controls – play, pause, stop, next, and previous track functionality.
* Track navigation – seamlessly switch between songs.
* Responsive UI – built using Tailwind CSS & Bootstrap for mobile and desktop compatibility.
* Routing – handled with React Router DOM.
* API ready – easily integrates with REST APIs (Axios used for requests).

# Architecture

## Component Structure

* App.jsx – root component, manages routes.
* Player.jsx – core music player UI & logic (controls, song info).
* Navbar.jsx – navigation bar for switching between sections.
* Playlist.jsx – displays a list of available tracks.
* Utilities – custom hooks & helper functions for state management and API calls.

## State Management

Local state: managed with useState & useEffect for track controls.  
Planned global state: React Context for playlist and user session management.

## Routing

Powered by React Router DOM (v6.20.1).  
Routes: /, /playlist, /player.

# Setup Instructions

## Prerequisites

Node.js >=16  
Package Manager: npm or yarn

## Installation

git clone <repo-url>  
cd music-player-frontend-  
npm install

## Environment Variables

If using a backend, configure .env with API URLs.

# Folder Structure

music-player-frontend-/  
│── public/ # Static assets  
│── src/  
│ ├── components/ # Reusable UI components  
│ ├── pages/ # Page-level components  
│ ├── assets/ # Images, icons  
│ ├── utils/ # Helper functions, hooks  
│ ├── App.jsx # Root component  
│ ├── main.jsx # Entry point  
│── index.html # Base HTML template  
│── package.json # Dependencies & scripts  
│── vite.config.js # Vite configuration

# Running the Application

Development server: npm run dev  
Production build: npm run build  
Preview build: npm run preview

# Component Documentation

## Player Component

Props: track, onPlay, onPause, onNext, onPrevious  
Function: Manages audio playback and updates UI controls.

## Playlist Component

Props: tracks (array of song objects)  
Function: Displays and allows track selection.

## Navbar Component

Provides navigation links using React Router DOM.

# User Interface & Styling

Frameworks: TailwindCSS (^3.3.6), Bootstrap (^5.3.2)  
UI Elements: React Icons for buttons & controls  
Responsive Design: Mobile-first layout

# Testing Strategy

Unit testing: Jest + React Testing Library  
Integration tests: focused on player controls  
Code coverage: not yet implemented

# Known Issues

No live backend integration (uses json-server for mock API)  
Limited error handling for missing or invalid tracks

# Future Enhancements

* Dark mode theming
* User authentication & custom playlists
* API integration with Spotify/YouTube
* Enhanced animations for UI transitions