Introduction:

Project Title :Rhythmic Tunes

Team Members:

- VARSHA B
- SHAKINA GRACY A
- RITHIKA DEVI D
- SELVABHARATHI S

Project overview:

Purpose:

Rhythmic Tunes is a modern and interactive music streaming application designed to provide users with an immersive and seamless experience in discovering, playing, and managing their favorite songs. The platform allows users to stream music, create personalized playlists, and explore trending tracks across different genres.

Rhythmic Tunes is designed to provide users with a seamless and engaging music streaming experience. The platform enables users to discover, play, and manage their favorite songs effortlessly. With an intuitive interface, personalized recommendations, and playlist management features, Rhythmic Tunes enhances the way users interact with music.

Rhythmic Tunes is developed to provide a modern, user-friendly, and immersive music streaming experience that caters to the needs of music lovers. The platform is designed to allow users to discover, stream, and manage their favorite songs with ease while ensuring a smooth and engaging interface.

Features:

1. **User Authentication & Personalization**
   - Secure Login & Signup – Users can register and log in securely using email or social accounts.
   - User Profiles – Personalized dashboard for managing playlists, liked songs, and preferences.
2. **Music Streaming & Playback**
   - High-Quality Audio Streaming – Supports multiple audio qualities for an optimized experience.
   - Playback Controls – Play, pause, skip, rewind, shuffle, and repeat functionalities.
   - Real-time Song Progress – Displays the current playback position with a progress bar.
3. **Smart Music Discovery & Search**
   - Advanced Search Bar – Allows users to search for songs, artists, albums, and genres.
   - Filters & Sorting – Users can sort by popularity, release date, and genre.
   - AI-Powered Recommendations – Suggests new songs and playlists based on user preferences.
   - Trending & Featured Playlists – Showcases top charts, new releases, and popular playlists.
4. **Playlist Creation & Management:**
   - Create & Customize Playlists – Users can create and name their playlists.
   - Add & Remove Songs – Easily manage songs within a playlist.
   - Auto-Generated Playlists – AI-generated recommendations based on user preferences.
   - Favorite & Liked Songs Section – Users can like songs and access them quickly.

Architecture :
The architecture of Rhythmic Tunes is designed to ensure scalability, efficiency, and seamless performance. It follows a modular and component-based approach using React.js for the frontend and a RESTful API for backend communication.

1. High-Level Architecture

- Client-Side (Frontend) – React.js
- Component-Based Architecture – Reusable UI components for better maintainability.
- State Management – Redux Toolkit is used for managing global state.
- Routing – Implemented using React Router for seamless navigation.
- UI Framework – Uses Tailwind CSS for styling and responsiveness.

2.Server-Side (Backend) – Node.js & Express.js

- REST API – Handles user authentication, song retrieval, and playlist management.
- Database – MongoDB (NoSQL) is used for storing user profiles, playlists, and song metadata.
- Authentication – JWT (JSON Web Token) is used for secure user authentication

3.Music Streaming & Storage

- Cloud Storage – Songs are stored in AWS S3 / Firebase Storage for optimized streaming.
- CDN (Content Delivery Network) – Ensures fast and efficient song streaming worldwide.

4.Third-Party Integrations

- Music APIs – Uses external APIs (e.g., Spotify API, Apple Music API) to fetch song metadata.

- Payment Gateway (for Premium Users) – Integrates Stripe/Razorpay for handling premium subscriptions.

State Management (Redux)

- **Global State:**
  userSlice.js – Manages user authentication and preferences.
  playlistSlice.js – Stores user-created playlists.
  playerSlice.js – Controls the currently playing song and playback status.
- **Local State:**
  Controlled using React useState for individual component UI updates.

Routing – Rhythmic Tunes

Routing in Rhythmic Tunes is implemented using React Router for the frontend and Express.js for the backend API. It enables smooth navigation between pages on the client side and structured API requests on the server side.

1. Frontend Routing (React Router)

Rhythmic Tunes uses React Router v6 for handling navigation across different pages. The routing structure ensures a seamless single-page application (SPA) experience without unnecessary page reloads.

2 Backend Routing (Express.js API)

The backend routes are structured using Express.js to handle API requests related to authentication, songs, playlists, and user profiles.

Prerequisites

Before installing the project, ensure you have the following installed on your system:

Node.js (v16+) – Download from nodejs.org
npm or yarn – Comes with Node.js installation
MongoDB – Install locally or use MongoDB Atlas
Git – Required for cloning the repository

Running the application:
To start the development server and run rhythmic tunes locally.

- Open the terminal and navigate to the project folder
- Run the following command
- npm start

This will start the react development server,and you can open the app in your browser at http://localhost:3000

Component documentation:

Key Components

1 Navbar Component
The Navbar component provides navigation links to different sections of the application, including Home, Search, and Profile. If a user is logged in, it displays their profile and logout option. Otherwise, it shows a login link.

2 Music Player Component
The Music Player component manages audio playback functionalities such as play, pause, skip, and volume control. It also displays the currently playing song along with its artist. The player allows seamless music streaming with background playback support.

3 Playlist Component
The Playlist component is responsible for displaying and managing user-created playlists. It lists all songs in a given playlist and provides options to

delete or modify the playlist. Users can create and customize playlists based on their preferences.

Reusable Components

1 Search Bar Component
The Search Bar component allows users to search for songs, albums, and artists. It takes user input and triggers a search function, displaying relevant results in real-time.
Music Player Test – Ensures play/pause/skip buttons function properly.

Search Functionality Test – Verifies search results are displayed accurately.

2 Song Card Component
The Song Card component is used to display individual songs with details such as title, artist, and album cover. It includes a play button that allows users to start streaming the song instantly.

3 Button Component
The Button component is a reusable UI element used throughout the application for actions like playing music, adding songs to playlists, and submitting forms. It ensures consistency in design and functionality.

Styling:

The styling of Rhythmic Tunes is designed to ensure a modern, responsive, and visually appealing user experience. The project uses a combination of CSS frameworks, pre-processors, and custom styles to enhance the UI/UX.

1. CSS Frameworks & Libraries Used

- Tailwind CSS – Used for utility-based styling, making development faster and maintaining consistency across the app.
- Styled-Components – Enables dynamic theming and component-based styling in React.
- CSS Modules – Helps in writing scoped styles for individual components to avoid conflicts.

Testing

Unit Testing

- Focuses on testing individual components and functions.
- Ensures that UI elements, Redux states, and API calls work as expected.
- Tools Used: Jest, React Testing Library

Integration Testing

- Tests how different components interact with each other (e.g., Search Bar with Song List).
- Ensures API responses are handled correctly.
- Tools Used: Jest, Axios Mock Adapter

End-to-End (E2E) Testing

- Simulates real-world user interactions (e.g., login, playing songs, creating playlists).
- Ensures the entire user journey works without issues.
- Tools Used: Cypress, Selenium

2. Test Coverage

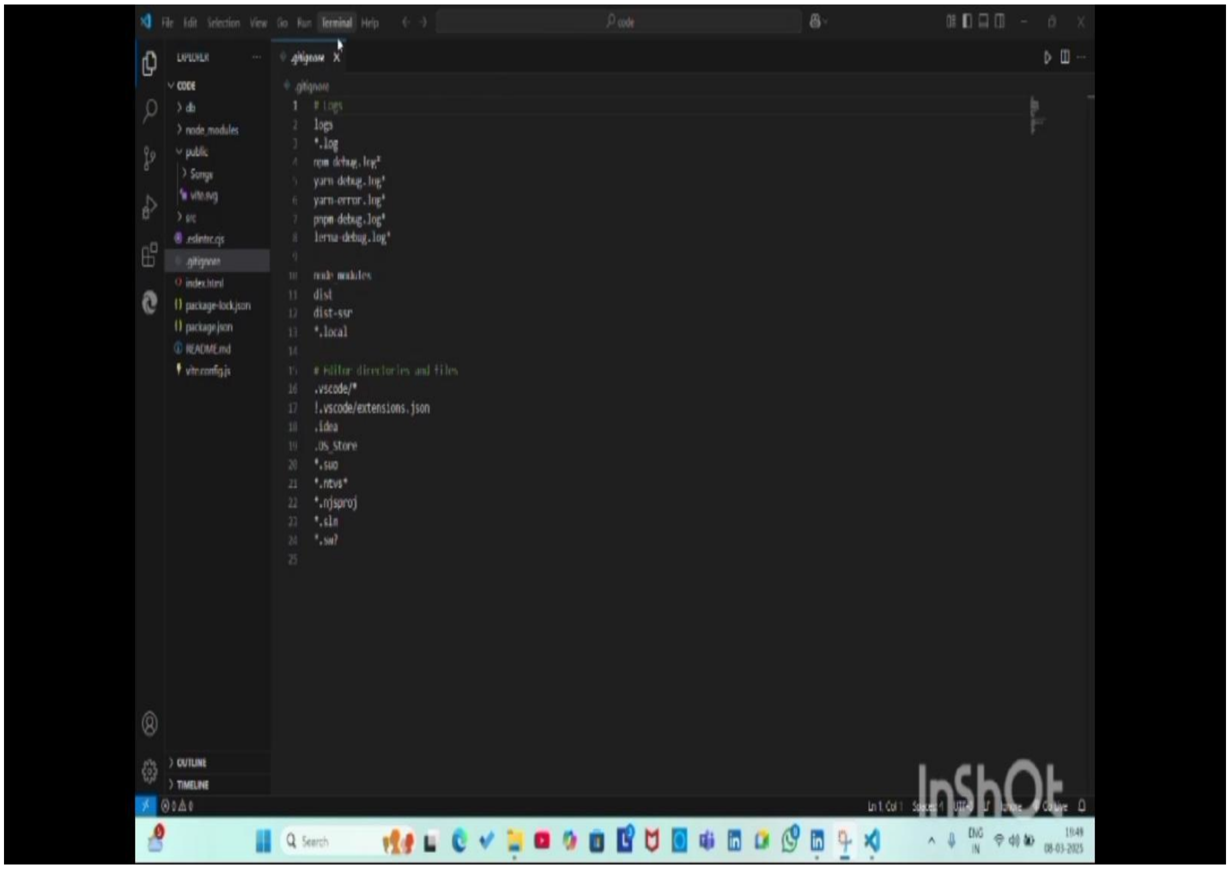
- Uses Jest Coverage Tool to track tested code sections.

- Ensures critical features like authentication, search, and playback are well-tested.
- Goal: 80%+ test coverage across components and API calls.

3. Types of Tests Conducted

- Frontend (React.js) Tests
- Navbar Component Test – Checks if navigation links work correctly.

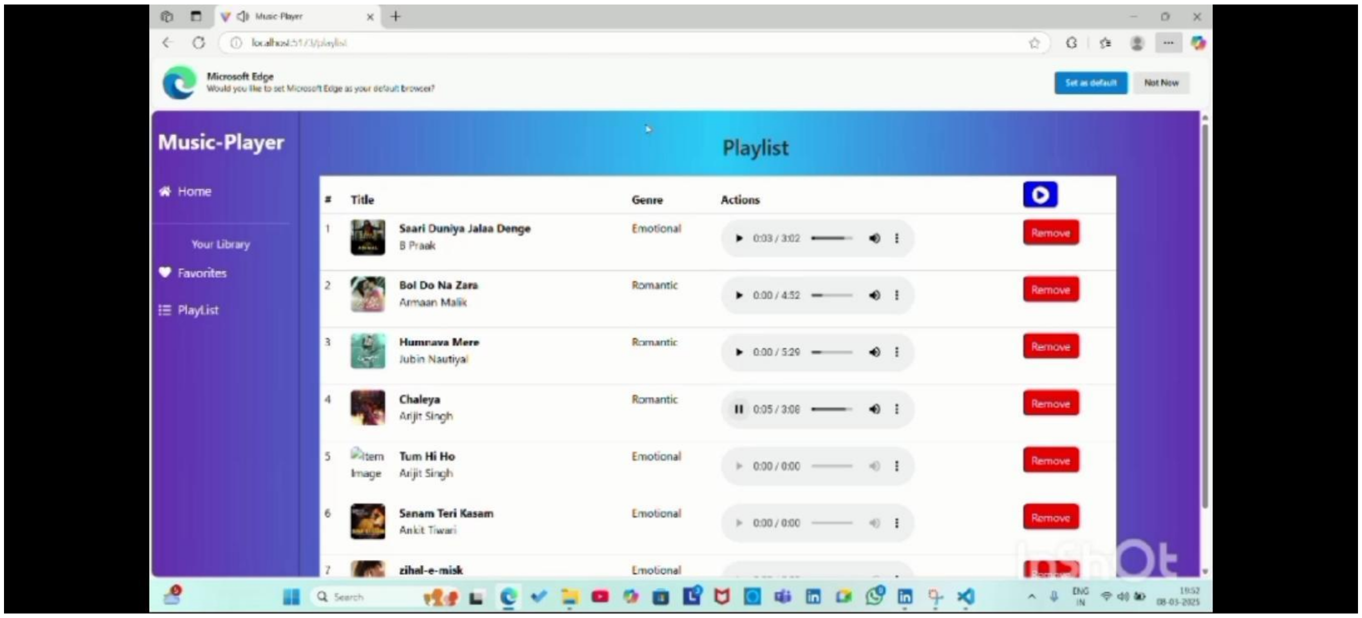
Screenshot and demo:



```
# Logs
logs
*.log
npm-debug.log*
yarn-debug.log*
yarn-error.log*
pnpm-debug.log*
lerna-debug.log*

node_modules
dist
dist-ssr
*.local

# Editor directories and files
.vscode/*
!.vscode/extensions.json
.idea
.DS_Store
*.suo
*.ntvs*
*.njsproj
*.sln
*.sw?
```



Playlist

#	Title	Genre	Actions	
1	Saari Duniya Jalaa Denge B Praak	Emotional	0:03 / 3:02	Remove
2	Bol Do Na Zara Armaan Malik	Romantic	0:00 / 4:52	Remove
3	Humnava Mere Jubin Nautiyal	Romantic	0:00 / 5:29	Remove
4	Chaleya Arijit Singh	Romantic	0:05 / 3:08	Remove
5	Item Image Tum Hi Ho Arijit Singh	Emotional	0:00 / 0:00	Remove
6	Senam Teri Kasam Ankit Tiwari	Emotional	0:00 / 0:00	Remove
7	zihal-e-misk	Emotional		

Future Enhancement:

Rhythmic Tunes is designed to deliver a seamless and engaging music streaming experience. However, several future enhancements can be implemented to improve user experience, performance, and scalability.

1. AI-Powered Recommendations

- Implement machine learning algorithms to analyze user behavior and provide personalized song recommendations.
- Use AI-based playlist generation to create dynamic playlists based on mood, genre, and listening habits.

2. Offline Mode (For Premium Users)

- Allow users to download songs for offline playback.
- Implement encrypted local storage to protect downloaded music files.

3. Social & Community Features

- Enable playlist sharing via social media.
- Allow users to follow friends and artists to see their listening activity.
- Add a commenting and rating system for albums and songs.

GitHub repository link:

https://github.com/Varshabalagi/RhythmicTunes
Demo video link:

https://drive.google.com/drive/folders/1D5ee4iUX8ZpPPXjj7AzXXsDIAb1dkiS7