**PROJECT REPORT**

Rhythmic Tunes :Your Melodic

Companion (React)

YEAR : **2024 – 2025**

COLLEGE NAME : **K.C.S KASI NADAR COLLEGE OF ARTS & SCIENCE**

CODE : UNM203

DEPARTMENT : **COMPUTER SCIENCE**

PROGRAM : **B.C.A**

SEMESTER **VI**

PROJECT SUBMITTED TO: UNIVERSITY OF MADRAS / NAAN MUDALVAN

Course Name : **Front End Development and Database Administration**

**TEAM LEADER: Dhanalakshmi. R**

**MEMBERS:**

1. NANDHINI .A.P

2. PREETHI .A

3. RAJESHWARI .S

4. PUJA GUCHHAIT.R

**GUIDED BY: MRS.R.PADMADEVI**

**SPOC NAME: Dr.K. LALITHAKAMESWARI**

**Rhythmic Tunes:** Your melodic companion

**(Music streaming application)**

**Introduction:**

Welcome to the future of musical indulgence – an unparalleled audio experience awaits you with our cutting-edge Music Streaming Application, meticulously crafted using the power of React.js. Seamlessly blending innovation with user-centric design, our application is set to redefine how you interact with and immerse yourself in the world of music.

Designed for the modern music enthusiast, our React-based Music Streaming Application offers a harmonious fusion of robust functionality and an intuitive user interface. From discovering the latest chart-toppers to rediscovering timeless classics,our platform ensures an all-encompassing musical journey tailored to your unique taste.

The heart of our Music Streaming Application lies in React, a dynamic and feature-rich JavaScript library. Immerse yourself in a visually stunning and interactive interface, where every click, scroll, and playlist creation feels like a musical revelation. Whether you're on a desktop, tablet, or smartphone, our responsive design ensures a consistent and enjoyable experience across all devices.

Say goodbye to the limitations of traditional music listening and welcome a world of possibilities with our React-based Music Streaming Application. Join us on this journey as we transform the way you connect with and savor the universal language of music.Get ready to elevate your auditory experience – it's time to press play on a new era of music streaming.

**Fractures of Rhythmic Tunes :**

* Song Listings: Display a comprehensive list of available songs with details such as title, artist, genre, and release date.
* Playlist Creation: Empower users to create personalized playlists, adding and organizing songs based on their preferences.
* Playback Control: Implement seamless playback control features, allowing users to play, pause, skip, and adjust volume during music playback.
* Offline Listening: Allow users to download songs for offline listening, enhancing the app's accessibility and convenience.
* Search Functionality: Implement a robust search feature for users to easily find specific songs, artists, or albums within the app.

**Sample Code For Fetching And Displaying Songs:**

import React, { useState, useEffect } from 'react';

import { Button, Form, InputGroup } from 'react-bootstrap';

import { FaHeart, FaRegHeart, FaSearch } from 'react-icons/fa';

import axios from 'axios';

import './sidebar.css'

function Songs() {

  const [items, setItems] = useState([]);

  const [wishlist, setWishlist] = useState([]);

  const [playlist, setPlaylist] = useState([]);

  const [currentlyPlaying, setCurrentlyPlaying] = useState(null);

  const [searchTerm, setSearchTerm] = useState('');

  useEffect(() => {

    // Fetch all items

    axios.get('http://localhost:3000/items')

      .then(response => setItems(response.data))

      .catch(error => console.error('Error fetching items: ', error));

  // Fetch favorities items

      axios.get('http://localhost:3000/favorities')

      .then(response => setWishlist(response.data))

      .catch(error => {

        console.error('Error fetching Favvorities:', error);

        // Initialize wishlist as an empty array in case of an error

        setWishlist([]);

      });

    // Fetch playlist items

    axios.get('http://localhost:3000/playlist')

      .then(response => setPlaylist(response.data))

      .catch(error => {

        console.error('Error fetching playlist: ', error);

        // Initialize playlist as an empty array in case of an error

        setPlaylist([]);

      });

      // Function to handle audio play

    const handleAudioPlay = (itemId, audioElement) => {

        if (currentlyPlaying && currentlyPlaying !== audioElement) {

          currentlyPlaying.pause(); // Pause the currently playing audio

        }

        setCurrentlyPlaying(audioElement); // Update the currently playing audio

      };

      // Event listener to handle audio play

      const handlePlay = (itemId, audioElement) => {

        audioElement.addEventListener('play', () => {

          handleAudioPlay(itemId, audioElement);

        });

      };

      // Add event listeners for each audio element

      items.forEach((item) => {

        const audioElement = document.getElementById(`audio-${item.id}`);

        if (audioElement) {

          handlePlay(item.id, audioElement);

        }

      });

      // Cleanup event listeners

      return () => {

        items.forEach((item) => {

          const audioElement = document.getElementById(`audio-${item.id}`);

          if (audioElement) {

            audioElement.removeEventListener('play', () => handleAudioPlay(item.id, audioElement));

          }

        });

      };

  }, [items,currentlyPlaying, searchTerm]);

  const addToWishlist = async (itemId) => {

    try {

      const selectedItem = items.find((item) => item.id === itemId);

      if (!selectedItem) {

        throw new Error('Selected item not found');

      }

      const { title, imgUrl, genre, songUrl, singer, id: itemId2 } = selectedItem;

      await axios.post('http://localhost:3000/favorities', { itemId: itemId2, title, imgUrl, genre, songUrl, singer });

      const response = await axios.get('http://localhost:3000/favorities');

      setWishlist(response.data);

    } catch (error) {

      console.error('Error adding item to wishlist: ', error);

    }

  };

  const removeFromWishlist = async (itemId) => {

    try {

      // Find the item in the wishlist by itemId

      const selectedItem = wishlist.find((item) => item.itemId === itemId);

      if (!selectedItem) {

        throw new Error('Selected item not found in wishlist');

      }

      // Make a DELETE request to remove the item from the wishlist

      await axios.delete(`http://localhost:3000/favorities/${selectedItem.id}`);

      // Refresh the wishlist items

      const response = await axios.get('http://localhost:3000/favorities');

      setWishlist(response.data);

    } catch (error) {

      console.error('Error removing item from wishlist: ', error);

    }

  };

  const isItemInWishlist = (itemId) => {

    return wishlist.some((item) => item.itemId === itemId);

  };

  const addToPlaylist = async (itemId) => {

    try {

      const selectedItem = items.find((item) => item.id === itemId);

      if (!selectedItem) {

        throw new Error('Selected item not found');

      }

      const { title, imgUrl, genre, songUrl, singer, id: itemId2 } = selectedItem;

      await axios.post('http://localhost:3000/playlist', { itemId: itemId2, title, imgUrl, genre, songUrl, singer });

      const response = await axios.get('http://localhost:3000/playlist');

      setPlaylist(response.data);

    } catch (error) {

      console.error('Error adding item to wishlist: ', error);

    }

  };

  const removeFromPlaylist = async (itemId) => {

    try {

      // Find the item in the wishlist by itemId

      const selectedItem = playlist.find((item) => item.itemId === itemId);

      if (!selectedItem) {

        throw new Error('Selected item not found in wishlist');

      }

      // Make a DELETE request to remove the item from the wishlist

      await axios.delete(`http://localhost:3000/playlist/${selectedItem.id}`);

      // Refresh the wishlist items

      const response = await axios.get('http://localhost:3000/playlist');

      setPlaylist(response.data);

    } catch (error) {

      console.error('Error removing item from wishlist: ', error);

    }

  };

  const isItemInPlaylist = (itemId) => {

    return playlist.some((item) => item.itemId === itemId);

  };

  const filteredItems = items.filter((item) => {

    const lowerCaseQuery = searchTerm.toLowerCase();

    return (

      item.title.toLowerCase().includes(lowerCaseQuery) ||

      item.singer.toLowerCase().includes(lowerCaseQuery) ||

      item.genre.toLowerCase().includes(lowerCaseQuery)

    );

  });

    return (

      <div style={{display:"flex", justifyContent:"flex-end"}}>

      <div className="songs-container" style={{width:"1300px"}}>

        <div className="container mx-auto p-3">

          <h2 className="text-3xl font-semibold mb-4 text-center">Songs List</h2>

          <InputGroup className="mb-3">

            <InputGroup.Text id="search-icon">

              <FaSearch />

            </InputGroup.Text>

            <Form.Control

              type="search"

              placeholder="Search by singer, genre, or song name"

              value={searchTerm}

              onChange={(e) => setSearchTerm(e.target.value)}

              className="search-input"

            />

          </InputGroup>

          <br />

          <div className="row row-cols-1 row-cols-md-2 row-cols-lg-3 row-cols-xl-4 g-4">

            {filteredItems.map((item, index) => (

              <div key={item.id} className="col">

                <div className="card h-100">

                  <img

                    src={item.imgUrl}

                    alt="Item Image"

                    className="card-img-top rounded-top"

                    style={{ height: '200px', width: '100%' }}

                  />

                  <div className="card-body">

                    <div className="d-flex justify-content-between align-items-center mb-2">

                      <h5 className="card-title">{item.title}</h5>

                      {isItemInWishlist(item.id) ? (

                        <Button

                          variant="light"

                          onClick={() => removeFromWishlist(item.id)}

                        >

                          <FaHeart color="red" />

                        </Button>

                      ) : (

                        <Button

                          variant="light"

                          onClick={() => addToWishlist(item.id)}

                        >

                          <FaRegHeart color="black" />

                        </Button>

                      )}

                    </div>

                    <p className="card-text">Genre: {item.genre}</p>

                    <p className="card-text">Singer: {item.singer}</p>

                    <audio controls className="w-100" id={`audio-${item.id}`} >

                      <source src={item.songUrl} />

                    </audio>

                  </div>

                  <div className="card-footer d-flex justify-content-center">

                    {isItemInPlaylist(item.id) ? (

                      <Button

                        variant="outline-secondary"

                        onClick={() => removeFromPlaylist(item.id)}

                      >

                        Remove From Playlist

                      </Button>

                    ) : (

                      <Button

                        variant="outline-primary"

                        onClick={() => addToPlaylist(item.id)}

                      >

                        Add to Playlist

                      </Button>

                    )}

                  </div>

                </div>

              </div>

            ))}

          </div>

        </div>

      </div>

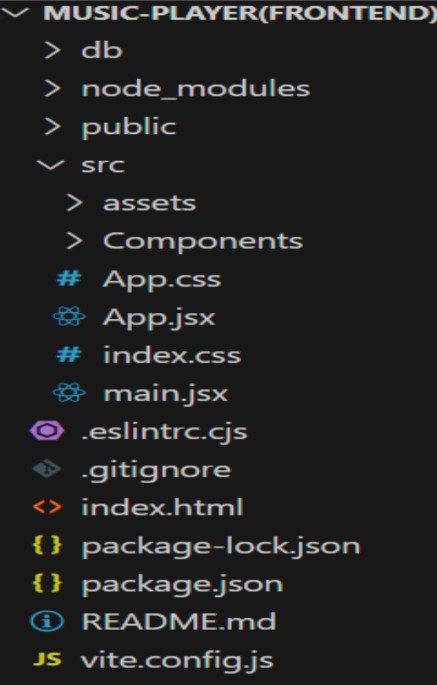
      </div>

    );

  }

export default Songs;

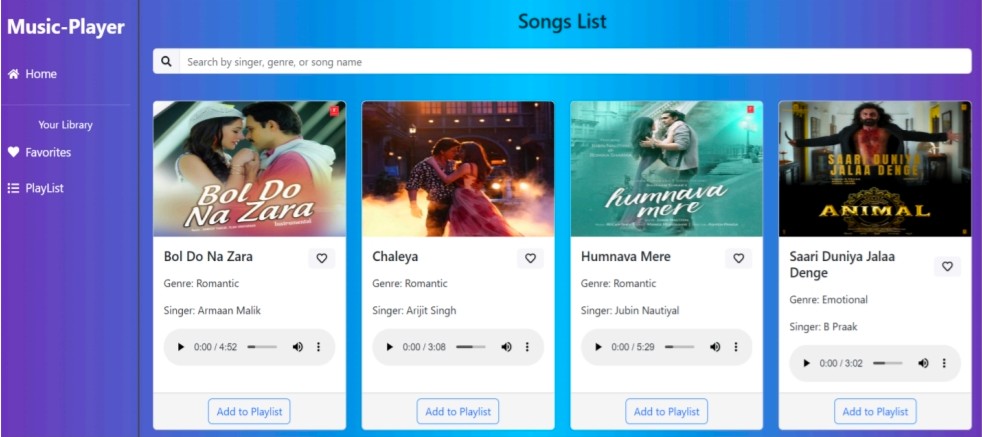
**Project structure:**

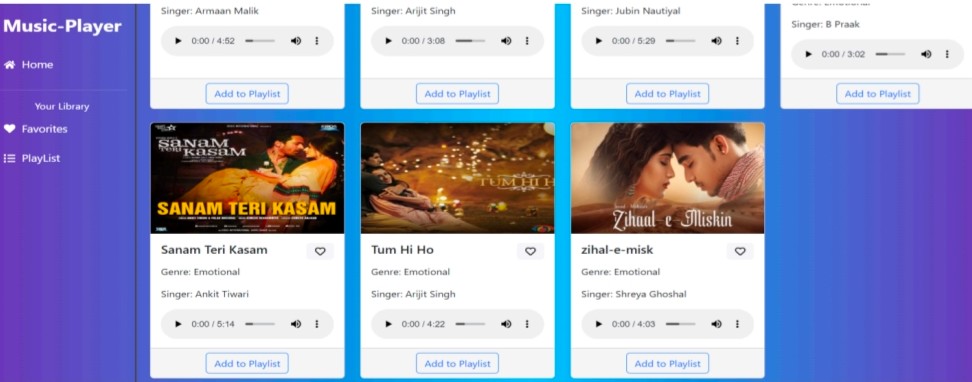
,

**Project Execution:**

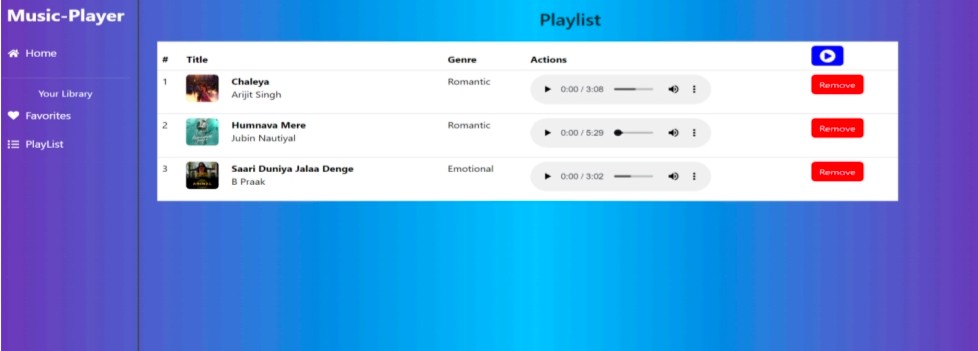
Here are some of the screenshots of the application.

* **Hero components**

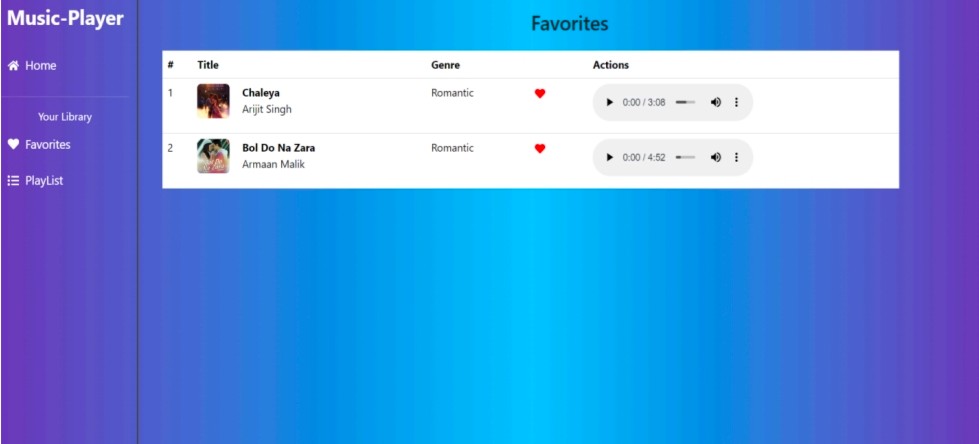
****

****

* **Playlist**

****

**Favourites**

****

Project Demo link:

<https://drive.google.com/file/d/1obzA_eHmSoaNOIB-oHNFCDJ08MVNhzxi/view?usp=drivesdk>

Project Source Code:

[dhanalakshmi2005/music-streaming-app](https://github.com/dhanalakshmi2005/music-streaming-app/tree/main)