RAM DHARSHAN E

ASSIGNMENT I

B.Tech AIDS-’B’

Ex.No.2 **Music Streaming Web Application**
Date:02.04.2025

Aim:
To develop a music streaming web application to provide users with a seamless and interactive music listening experience.

Procedure:

# Music Player Application

A modern, responsive music player application built with React, TypeScript, and Tailwind CSS.

## Features

### Authentication

- User login system with local storage persistence

- Session management with automatic login state restoration

- Secure logout functionality

### Music Library

- Multiple playlist support

- Song categorization by playlists

- Rich media display with album artwork

- Detailed song information (title, artist, duration)

### Playback Controls

- Play/Pause functionality

- Progress bar with seek capability

- Volume control with mute/unmute

- Continuous playback within playlists

- Visual feedback for currently playing track

### User Interface

- Responsive design for all screen sizes

- Beautiful, modern UI with smooth transitions

- Intuitive navigation between playlists

- Visual indicators for active states

- Hover effects for interactive elements

## Technical Architecture

### Component Structure

```
src/
├── components/
│   ├── Dashboard.tsx     # Main application layout
│   ├── Login.tsx         # Authentication screen
│   ├── MusicPlayer.tsx  # Audio player controls
│   ├── PlaylistCard.tsx # Playlist selection cards
│   └── SongList.tsx      # Song listing component
├── context/
│   └── AuthContext.tsx  # Authentication state management
└── types.ts              # TypeScript interfaces
```

### Key Technologies

- **React 18**: Modern UI development

- **TypeScript**: Type-safe development

- **Tailwind CSS**: Utility-first styling

- **Lucide React**: Icon system

- **HTML5 Audio API**: Native audio playback
- **Local Storage**: State persistence

## Data Structure

### User Interface
```typescript
interface User {
  username: string;
  password: string;
}
```

### Music Data
```typescript
interface Song {
  id: string;
  title: string;
  artist: string;
  duration: string;
  coverUrl: string;
  audioUrl: string;
}

interface Playlist {
  id: string;
  name: string;
  songs: Song[];
}
```

~~~

Usage Guide

Authentication
1. Launch the application
2. Enter any username and password on the login screen
3. Click "Sign In" to access the main dashboard

Playing Music
1. Select a playlist from the sidebar
2. Click on any song to start playback
3. Use the bottom player controls to:
   - Play/Pause the current track
   - Adjust playback position using the progress bar
   - Toggle audio mute
   - View current track information

Managing Playlists
- View all playlists in the left sidebar
- Click on a playlist to view its songs
- Each playlist shows the number of contained songs

Development Setup

1. Install dependencies:
```bash
npm install
``
~~~

~~~
2. Start development server:

```bash

npm run dev

```

3. Build for production:

```bash

npm run build

``

## Project Structure

```

/

├── src/   # Source code

├── public/   # Static assets

├── index.html   # Entry HTML

├── package.json   # Dependencies

├── tsconfig.json   # TypeScript config

├── vite.config.ts   # Vite config

└── tailwind.config.js # Tailwind CSS config

```
~~~

**Program:**

***App.jsx***

```jsx
import React from 'react';
import { AuthProvider, useAuth } from './context/AuthContext';
import { Login } from './components/Login';
import { Dashboard } from './components/Dashboard';

function AppContent() {
```

```
  const { user } = useAuth();
  return user ? <Dashboard /> : <Login />;
}

function App() {
  return (
    <AuthProvider>
      <AppContent />
    </AuthProvider>
 );}
export default App;
```

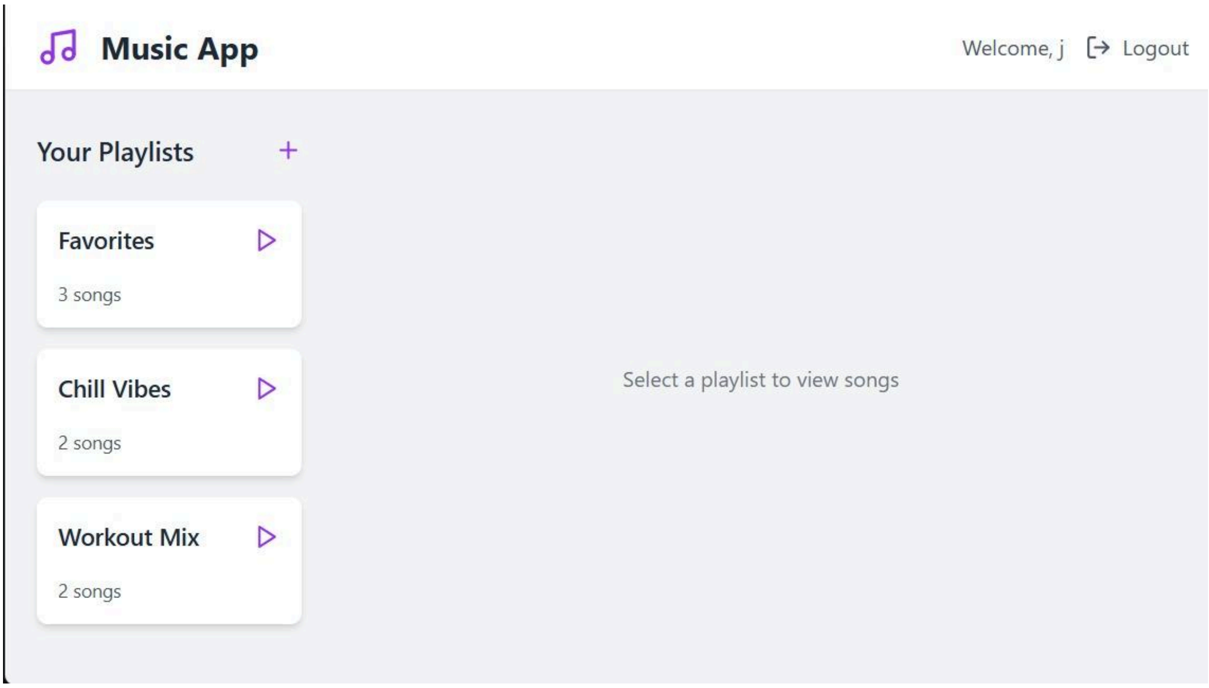
music.jsx

```
import React, { useState } from 'react';
import { useAuth } from '../context/AuthContext';
import { PlaylistCard } from './PlaylistCard';
import { SongList } from './SongList';
import { MusicPlayer } from './MusicPlayer.tsx';
import type { Playlist, Song } from '../types';
import { Music, LogOut, Plus } from 'lucide-react';

const SAMPLE_PLAYLISTS: Playlist[] = [
  {
    id: '1',
    name: 'Favorites',
    songs: [
      {
        id: '1',
        title: 'Summer Breeze',
        artist: 'Lofi Dreamer',
        duration: '3:24',
        coverUrl:
'https://images.unsplash.com/photo-1514924013411-cbf25faa35bb?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-summer-fun-13.mp3'
      },
      {
        id: '2',
        title: 'Sunset Memories',
        artist: 'Chill Beats',
        duration: '2:45',
        coverUrl:
'https://images.unsplash.com/photo-1511671782779-c97d3d27a1d4?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-tech-house-vibes-130.mp3'
      },
      {
        id: '3',
        title: 'Urban Dreams',
        artist: 'City Lights',
        duration: '3:15',
        coverUrl:
'https://images.unsplash.com/photo-1471478331149-c72f17e33c73?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-hip-hop-02-738.mp3'
      }
    ]
  },
  {
    id: '2',
    name: 'Chill Vibes',
    songs: [
      {
        id: '4',
        title: 'Ocean Waves',
        artist: 'Nature Sounds',
        duration: '4:20',
        coverUrl:
'https://images.unsplash.com/photo-1507525428034-b723cf961d3e?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-relaxing-in-nature-522.mp3'
      },
      {
        id: '5',
        title: 'Midnight Jazz',
        artist: 'Jazz Ensemble',
        duration: '3:50',
        coverUrl:
'https://images.unsplash.com/photo-1415201364774-f6f0bb35f28f?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-jazz-bar-3.mp3'
      }
    ]
  },
  {
    id: '3',
    name: 'Workout Mix',
    songs: [
      {
        id: '6',
        title: 'Power Up',
        artist: 'Energy Beats',
        duration: '3:30',
        coverUrl:
'https://images.unsplash.com/photo-1534258936925-c58bed479fcb?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-sport-rock-01-729.mp3'
      },
      {
        id: '7',
        title: 'Running High',
        artist: 'Fitness Crew',
        duration: '2:55',
        coverUrl: 'https://images.unsplash.com/photo-1486218119243-13883505764c?w=400&q=80',
        audioUrl: 'https://assets.mixkit.co/music/preview/mixkit-fitness-motivator-206.mp3'
      }
    ]
  }
];

export const Dashboard: React.FC = () => {
  const { user, logout } = useAuth();
  const [playlists] = useState<Playlist[]>(SAMPLE_PLAYLISTS);
  const [selectedPlaylist, setSelectedPlaylist] = useState<Playlist | null>(null);
  const [currentSong, setCurrentSong] = useState<Song | null>(null);
  const [isPlaying, setIsPlaying] = useState(false);

  const handlePlaySong = (song: Song) => {
    setCurrentSong(song);
    setIsPlaying(true);
  };

  const handleTogglePlay = () => {
    setIsPlaying(!isPlaying);
```

```
};

return (
  <div className="min-h-screen bg-gray-100">
    <header className="bg-white shadow-sm">
      <div className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-4">
        <div className="flex items-center justify-between">
          <div className="flex items-center space-x-4">
            <Music className="w-8 h-8 text-purple-600" />
            <h1 className="text-2xl font-bold text-gray-800">Music App</h1>
          </div>
          <div className="flex items-center space-x-4">
            <span className="text-gray-600">Welcome, {user?.username}</span>
            <button
              onClick={logout}
              className="flex items-center space-x-2 text-gray-600 hover:text-gray-800"
            >
              <LogOut className="w-5 h-5" />
              <span>Logout</span>
            </button>
          </div>
        </div>
      </div>
    </header>

    <main className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-8">
      <div className="grid grid-cols-1 md:grid-cols-4 gap-6">
        <div className="md:col-span-1 space-y-6">
          <div className="flex items-center justify-between">
    <h2 className="text-xl font-semibold text-gray-800">Your Playlists</h2>
    <button className="text-purple-600 hover:text-purple-700">
      <Plus className="w-5 h-5" />
    </button>
  </div>
  <div className="space-y-4">
    {playlists.map((playlist) => (
      <PlaylistCard
        key={playlist.id}
        playlist={playlist}
        onSelect={setSelectedPlaylist}
      />
    ))}
  </div>
</div>

<div className="md:col-span-3">
  {selectedPlaylist ? (
    <div className="bg-white rounded-lg shadow-sm p-6">
      <h2 className="text-2xl font-bold text-gray-800 mb-6">
        {selectedPlaylist.name}
      </h2>
      <SongList
        songs={selectedPlaylist.songs}
        onPlay={handlePlaySong}
        currentSong={currentSong}
        isPlaying={isPlaying}
      />
    </div>
```

```
            ) : (
              <div className="flex items-center justify-center h-full">
                <p className="text-gray-500">Select a playlist to view songs</p>
              </div>
            )}
          </div>
        </div>

        {currentSong && (
          <MusicPlayer
            song={currentSong}
            isPlaying={isPlaying}
            onTogglePlay={handleTogglePlay}
          />
        )}
      </main>
    </div>
  );
};
```

Output:



Select a playlist to view songs

Result:

Thus, the music streaming web application to provide users with a seamless and interactive music listening experience was developed and the output was verified.