PROJECT REPORT

on

"SoundSphere"

Submitted to

Rashtrasant Tukdoji Maharaj Nagpur University, Nagpur

In Partial Fulfillment of the requirement of

Master of Commerce (Computer Management)

SEM-IV

Submitted by

Prachi Burde
Manjil Pillewan

Under the Guidance of

Dr. Sarang Javkhedkar

Dr. Ambedkar Institute of Management Studies & Research Deeksha Bhoomi Nagpur-440012

(2024-2025)

CERTIFICATE

This is to certify that **Prachi Burde, Manjil Pillewan** have satisfactorily completed the project work entitled **"SoundSphere"** in less than one academic session. This also certify that this project work is the result of the candidate's own work and is of sufficiently high standard to warrant its presentation for the M.Com. (Computer Management) program.

To the best of my knowledge, this project or its part has not been submitted to this university or any other university for any Degree/Diploma.

Guide Signature

(Dr. Sarang Javkhedkar)

Internal Examiner **External Examiner**

Place: Nagpur

Date: **Director**

DECLARATION

We, Prachi Burde, Manjil Pillewan hereby declare that the project entitled **“**SoundSphere**”** is the outcome of our own research work based on personal study during academic session 2024– 2025 and has not been submitted previously for award of any degree or diploma to this university or any other university.

Prachi Burde

Manjil Pillewan

ACKNOWLEDGEMENT

"Words have never expressed human sentiments. This only an attempt to express our deep gratitude which comes from our heart."

It is a great pleasure for us to express our deep feeling of gratitude to our respected guide **Dr. Sarang Javkhedkar (Asst. Professor, Dr. Ambedkar Institute of Management Studies & Research, Nagpur)** for his great encouragement and unfailing support which provided needed moral and confidence to carry on our work.

We are grateful to the **Dr. S. S. Fulzele**, (**Director, Dr. Ambedkar Institute of Management Studies & Research, Nagpur)** for making all facilities available for our work.

It is with profound gratitude that we wish to express our indebtedness to **Dr. Nirzar Kulkarni** (**Associate Director, Dr. Ambedkar Institute of Management Studies & Research, Nagpur**) for his invaluable guidance and supervision in completion of this project work.

We are grateful to our parents for their lovable support. Last but not the least we are thankful to our friends and other faculty member for their direct and indirect help for completion of this work.

Content Page

1. Introduction
2. Objective
3. Preliminary System Analysis
    - Preliminary Investigation.
    - Present System in Use.
    - Flaws in Present System.
    - Need of New System.
    - Feasibility Study.
    - Project Category
4. Software Engineering Paradigm Applied
    - Modules
    - System / Modular Chart.
5. Software & Hardware Requirement Specification.
6. Detailed System Analysis.
    - Data Flow Diagram.
    - Numbers of Modules and Process Logic.
    - Data Structures and Tables.
    - Entity-Relationship Diagram.
7. System Design.
    - Page Design.
    - Source Code.
    - Input screen & Output Screen.
8. System Security Measures.
9. Implementation, Evaluation and Maintenance.
10. Future Scope of the project.
11. Suggestion & Conclusion
12. Bibliography& References.

1. Introduction

In the rapidly evolving digital age, the way people consume music has undergone a dramatic transformation. Music streaming platforms have become the cornerstone of modern music consumption, offering instant access to millions of tracks with just a few clicks. These platforms have not only changed user behavior but also reshaped the music industry itself, offering artists a new medium to reach global audiences.

SoundSphere seeks to fill this void. It is a **web-based music streaming platform** thoughtfully crafted to celebrate the timeless essence of music from the **60s, 70s, 80s, 90s, and early 2000s**. The core objective of SoundSphere is to offer users a seamless and engaging experience while exploring musical gems from different eras. Whether it's rediscovering legendary artists, revisiting golden hits, or diving deep into the evolution of musical genres—SoundSphere serves as a digital time capsule for music lovers.

The platform offers:

- A **clean and user-friendly interface** that encourages easy navigation.
- **Genre-based filtering** to help users find music according to their preferences.
- **Artist showcases** and detailed artist sections to highlight musical journeys.
- **Interactive music popups** and playlist features for immersive playback.
- **Personalized recommendations and subscriptions** to keep users engaged and updated.

SoundSphere is built using modern web technologies to ensure optimal performance, responsiveness, and accessibility across various devices. Its modular architecture not only allows for smooth functionality but also lays a scalable foundation for future enhancements such as mobile integration, user profiles, and AI-based music suggestions.

Ultimately, **SoundSphere bridges the gap between the past and the present**, blending the emotional value of retro music with the convenience of modern digital tools. It caters not just to the nostalgic listener but also to younger generations curious about the roots of the music they enjoy today. By harmonizing innovation with tradition, SoundSphere stands as a unique and promising addition to the digital music ecosystem.

2. Objectives

The main objectives of the **SoundSphere** project are as follows:

1. **To develop a web-based music streaming platform** that allows users to play songs from different decades seamlessly.

2. **To enhance user experience** by integrating a responsive and interactive UI with features such as search, playlists, and album browsing.

3. **To provide an extensive music library** categorized by eras (60s, 70s, 80s, 90s, 2000s) to help users explore historical music trends.

4. **To implement a dynamic music player** with essential playback controls, including play, pause, next, previous, and volume control.

5. **To ensure compatibility across devices**, making the platform accessible on desktops, tablets, and mobile phones.

6. **To optimize performance and efficiency** by using lightweight web technologies to deliver smooth streaming without significant buffering.

7. **To incorporate social media integration**, enabling users to share their favorite songs and playlists with friends.

8. **To maintain data security and privacy** by ensuring that user information and preferences are protected.

3. Preliminary System Analysis

i) Preliminary Investigation

Preliminary investigation is the initial phase where the project's scope, objectives, and requirements are identified. It helps in understanding the problem space, analyzing potential solutions, and determining if the proposed project is feasible.

Objective of the Project:
The main objective of developing SoundSphere is to create a retro-themed music player dedicated to classic Bollywood songs from the '60s, '70s, and '80s. The aim is to offer a nostalgic, user-friendly experience that enables users to access curated playlists of timeless melodies.

Origin of the Project:
The idea emerged from the increasing interest in retro content, especially among people who cherish the golden era of Bollywood music. Unlike mainstream platforms that cater to a broad audience, this project focuses on providing a culturally relevant and immersive experience.

Target Audience:

- Music enthusiasts who appreciate classic Bollywood music.
- Older generations who grew up listening to these timeless tracks.
- Younger audiences interested in exploring retro music for cultural knowledge.
- Content creators and influencers interested in nostalgic content.

ii) Present System in Use

The present systems in use for listening to Bollywood retro music include popular streaming platforms like Spotify, Apple Music, YouTube, and regional platforms like Gaana and JioSaavn.

Advantages of Current Systems:

- Extensive music libraries covering a wide range of genres and languages.
- Advanced recommendation systems powered by AI.
- User-friendly mobile and desktop applications.
- Offline access for premium users.

Challenges with Current Systems:

- The overwhelming variety of music makes it difficult to find decade-specific, curated retro playlists.
- The interface is modern and lacks a nostalgic touch, diminishing the retro experience.
- Mixed recommendations of modern and retro music, diluting the classic listening experience.
- Difficulty in finding accurate metadata and contextual information about classic tracks.

iii) Flaws in Present System

While current music streaming platforms are advanced, they have limitations when it comes to catering to niche, retro-themed audiences. Here's a detailed look at the flaws:

1. **Overloaded Content:**
    - The vast catalog of music makes it hard to locate classic tracks, especially if users don't remember exact titles.
    - Retro Bollywood fans often face difficulty in filtering out modern music from their search results.

2. **Lack of Curation:**
    - There is a lack of carefully curated playlists that understand the cultural essence of different Bollywood eras.
    - Generic playlists often miss out on iconic but lesser-known tracks.

3. **Generic Interface:**
    - Existing platforms have modern, sleek interfaces that do not resonate with the retro theme.
    - There is no thematic visualization that captures the aesthetic of Bollywood's golden era.

4. **Unfocused Recommendations:**
    - Music recommendation algorithms prioritize trending music over classic, nostalgic tracks.
    - Retro listeners often experience unwanted suggestions of contemporary music.

5. **Limited Offline Access:**
    - While offline access exists, it is usually part of a paid plan, making it inaccessible for some users.
    - Users interested in classic music may not wish to pay for premium access solely for offline features.

iv) Need of New System

Considering the flaws in the present systems, there is a clear need for a dedicated retro Bollywood music platform. **SoundSphere** aims to address these gaps by providing:

1. **Focused Curation:**
    - Carefully curated playlists that represent specific decades, movies, or legendary artists like Rafi, Lata, Kishore, and Mukesh.
    - Categorization of playlists by theme — romantic classics, iconic duets, patriotic songs, etc.

2. **Enhanced User Experience:**
    - A retro-inspired interface with elements reminiscent of vintage Bollywood posters and aesthetics.
    - Interactive features like trivia, fun facts about songs, and singer biographies to enrich the user experience.

3. **Efficient Search and Discovery:**
    - Search functionality optimized for finding classic tracks based on movie name, actor, or music director.
    - Advanced filters that allow users to search by decade or genre of classic Bollywood music.

4. **Platform Accessibility:**
    - Availability across devices (desktop, mobile, tablet) ensuring ease of access.
    - Compatibility with different operating systems like Windows, macOS, Android, and iOS.

5. **Cultural Resonance:**
    - Highlighting the significance of Bollywood's golden era, creating a sense of nostalgia.
    - Involvement of a community where fans can share memories or discuss their favorite retro songs.

v) Feasibility Study

A feasibility study helps in assessing whether developing **SoundSphere** is viable in terms of technical, economic, and operational aspects.

1. Technical Feasibility:

- **Technology Stack:** Frontend using **HTML**, **CSS**, and **JavaScript**; Backend using **PHP** or **MySQL** for data handling.
- **Music Database:** Use of public music APIs or licensed databases for music data.
- **Compatibility:** Cross-browser compatibility and responsive design using **Bootstrap**.
- **Development Tools:** Use of development environments like Visual Studio Code and collaboration tools like Git.

2. Economic Feasibility:

- **Cost Estimation:**
  - **Development:** Primarily time and effort if managed by a small team.
  - **Hosting and Maintenance**: Costs for server hosting and domain registration.
  - **Licensing:** Potential copyright costs for music streaming.
- **Revenue Model:**
  - Advertisements for non-premium users.
  - Subscription-based premium plan for ad-free and offline access.

3. Operational Feasibility:

- **User-Friendliness:** Easy-to-navigate UI suitable for all age groups.
- **Community Engagement:** Retro fans and communities can help promote the platform.
- **Maintenance and Support:** Periodic updates, bug fixes, and adding new playlists to keep the platform engaging.

vi) Project Category

This project falls under the category of **Web Application Development** with a specific focus on **Digital Media Streaming** and **Entertainment Technology**.

- **Digital Media Streaming:** The core function is streaming retro Bollywood music.
- **Entertainment Technology:** Providing an enjoyable, culturally rich experience to users.
- **Web Application Development:** Leveraging modern web development practices to build a responsive, accessible, and nostalgic platform.

4. Software Engineering Paradigm Applied

The development of SoundSphere follows a structured approach to ensure systematic planning, efficient design, and smooth implementation. The chosen paradigm for this project is the Modular Design Approach combined with elements of the Waterfall Model for clear sequential development.

Software Engineering Paradigm
Waterfall Model with Modular Design Approach:

The Waterfall Model is a linear, sequential approach where each phase of development depends on the completion of the previous phase. This approach is suitable for this project because the requirements are well understood, and the scope is clearly defined. The Modular Design Approach further enhances it by dividing the application into self-contained, manageable modules, ensuring each part has a clear responsibility.

i) Modules

The **SoundSphere** project is built on a modular software design paradigm, breaking the system into manageable, functional components that interact with each other efficiently. The following modules were identified and implemented based on the system's objectives and user interaction flow:

1. Home Module (home.html)
   - Acts as the landing page for users.
   - Includes the **Main Cover** with a "Start Listening" call-to-action.
   - Offers **Genre-based filtering** to help users discover music easily.
   - Features **music cards with popup song lists** to preview and select songs.

2. Featured Module (#featured)
   - Displays specially curated playlists or highlighted songs.
   - Integrates **popup song list** functionality for instant playback.
   - Interacts with the database to fetch up-to-date featured content dynamically.

3. Retro Tracks Module (retro.html)
   - Dedicated to old-school and 90s music content.
   - Includes a **Retro Cover** section for thematic visual appeal.
   - Allows filtering via **genre buttons**.
   - Supports **Top 90's Tracks with Popups** and **Featured Playlists** with a built-in music player.

4. Artists Module (artists.html)

- Enables browsing through artists using **Filter Buttons**.
- Each artist card opens a **Popup List with songs** and an embedded player.
- Provides dynamic interaction and direct playback for a smoother user experience.

5. Subscription Module

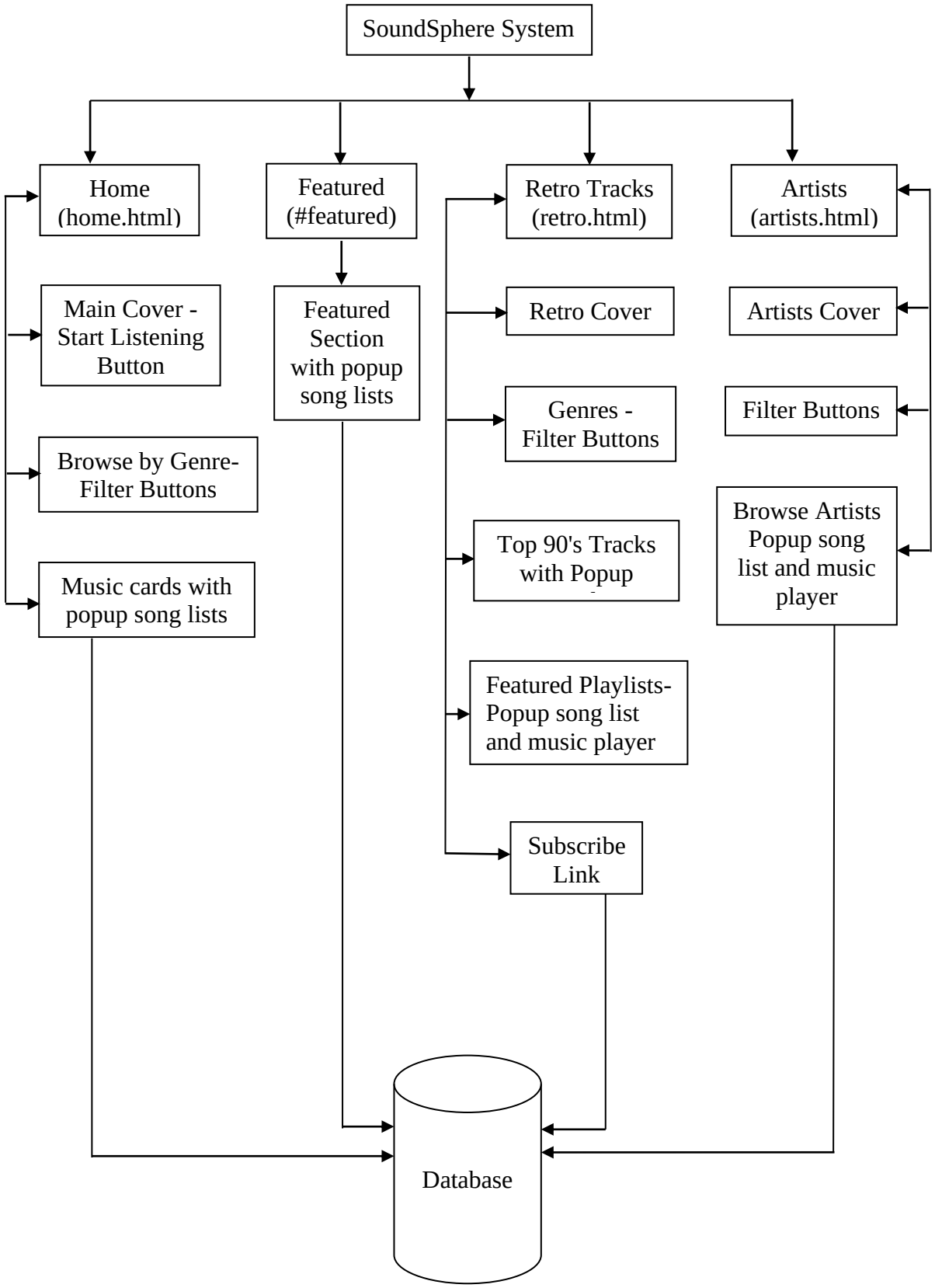
- Allows users to **subscribe** to updates via a **Subscribe Link**.
- Stores **ID, Email, and Time-Date** in the database.
- Integrated with a popup notification system confirming successful subscriptions.

6. Database Module

- Centralized data handling for music tracks, featured lists, artists, and subscriber data.
- Facilitates read/write operations between all modules.
- Ensures content integrity and supports scalable access to audio and metadata.

This modular architecture enhances the maintainability, scalability, and efficiency of the **SoundSphere** system. Each module is independently manageable, reusable, and can evolve without affecting the rest of the system.

ii) System / Modular Chart.



Explanation of Modular Chart:

1. Home Module (home.html):

- Acts as the main **UI Module**, where users first land.
- Provides **genre-based filtering** and displays **music cards** with popup song lists.
- Fetches data from the **Music Database Module** to display song lists dynamically.

2. **Featured Module (#featured):**

- Works as a **curated music section**, displaying trending and recommended songs.
- Contains a popup song list fetched from the **Music Database Module**.
- Integrated into the **UI Module** for seamless navigation.

3. Retro Tracks Module (retro.html):

- Specializes in displaying **retro music**, filtering by genres and featuring **top 90s tracks**.
- Includes **featured playlists** with a popup song list and **music playback functionality**.
- Connected to the **Music Database Module** to retrieve song information.
- Also linked to the **Subscribe Link** for user engagement.

4. Artists Module (artists.html):

- Allows users to browse songs by artists using **filter buttons**.
- Displays an **artist cover section** and a **popup song list with a music player**.
- Pulls artist and song data from the **Music Database Module**.

5. Music Database Module:

- Serves as the **backend data hub**, storing all music, artist details, and metadata.
- Supplies content to **Home, Featured, Retro Tracks, and Artists Modules**.
- Ensures smooth streaming and data retrieval for the music player.

6. Authentication & Admin Module (Not shown in the chart but essential):

- Controls **subscription handling**, allowing users to sign up for updates.
- Manages content security, song additions, and database maintenance.
- Oversees the **Featured Section** and ensures proper categorization in all modules.

Software & Hardware Requirement Specification

Software Requirements

Frontend Technologies:

- **HTML5 & CSS3** – For structuring and designing the user interface.
- **JavaScript (ES6+)** – For interactive features, animations, and event handling.
- **React.js** – If using a framework for dynamic rendering.

Backend Technologies:

- **PHP (or Node.js)** – To handle backend logic and database connections.
- **MySQL** – For storing user data, songs, and metadata.
- **Apache** – Web server to host the backend application.

Music Playback & Streaming Technologies:

- **HTML5 Audio API** – To handle music playback within the browser.
- **Media Streaming Protocols (MP3 streaming)** – For smooth audio playback.

Development & Testing Tools:

- **VS Code / Sublime Text / JetBrains WebStorm** – For coding and debugging.
- **Git & GitHub/GitLab/Bitbucket** – For version control and collaboration.
- **XAMPP / LAMP Stack (For PHP & MySQL locally)** – If using PHP backend.

Hardware Requirements

Minimum System Requirements (For Development & Hosting Server):

- **Processor:** Intel Core i5 or AMD Ryzen 5 (or better).
- **RAM:** Minimum **8GB RAM** (16GB recommended for seamless development).
- **Storage:** At least **256GB SSD** (512GB SSD recommended for faster performance).
- **Sound Card & Speakers:** Required for audio testing and playback.

Server Specifications (For Live Hosting):

- **Processor:** Multi-core CPU (Intel Xeon or AMD EPYC).
- **RAM:** Minimum **4GB** (8GB recommended for better performance).
- **Storage:** SSD-based storage (50GB+ for song storage).
- **Network Bandwidth:** High-speed internet (1Gbps or higher for smooth streaming).

User-End System Requirements (For Smooth Playback):

- **Device:** Desktop, Laptop or Smartphone.
- **Browser:** Chrome, Edge(Latest Versions).
- **Internet Speed:** Minimum **3Mbps for normal streaming**, **10Mbps+ for HD audio**.

Detailed System Analysis

Detailed system analysis provides a structured overview of how data flows within the **SoundSphere Music System**, the modules involved, database structures, and the relationships between different entities.

i) Data Flow Diagram

A **Data Flow Diagram (DFD)** visually represents how data moves through the system. Below is an overview of the **DFD for SoundSphere** :

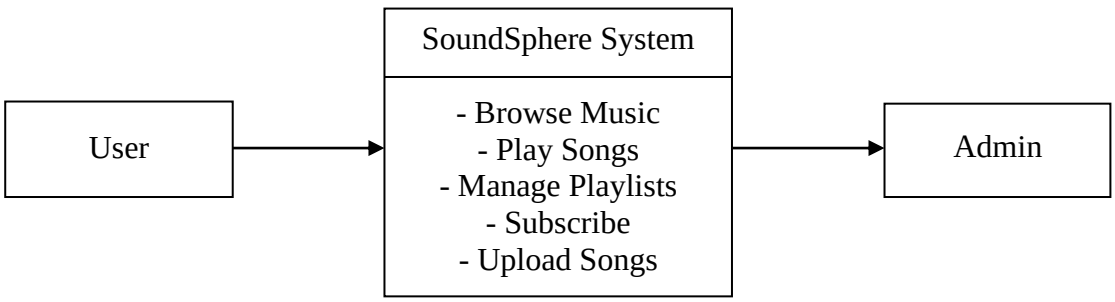
DFD Level 0 (Context Diagram) – SoundSphere System
This is the highest level representation of the system, showing the main interactions between the system and external entities (Users & Admins).

Entities & Flow:

1. **User** interacts with the **SoundSphere System**
   - Browse Music
   - Play Songs
   - Create Playlists
   - Subscribe for Updates

2. **Admin** manages the system
   - Uploads songs
   - Manages playlists
   - Monitors user activity

Diagram Representation:



ii) Numbers of Modules and Process Logic

Numbers of Modules

1. **Front-End (User Interface):**
   - The user interacts with the **Home, Featured, Retro Tracks, and Artists** pages.
   - The UI is designed using **HTML, CSS, and JavaScript**.
   - Optional JavaScript validation is used for search functionality, music player interactions, and subscription form validation**.**

2. **Back-End(Server Processing):**
   - User actions (like playing a song, filtering music, subscribing to the newsletter) send requests to the server.
   - PHP processes these requests, ensuring valid data is retrieved and stored.
   - Example:
     - If a user subscribes, the email is sent via POST to subscribe.php.

3. **Database (Music & Users Data Storage)**:
   - **Music Data** is stored in a database (**MySQL**).
   - Tables include:
     - `songs` → Stores song details (title, artist, genre, duration).
     - `users` → Stores registered users (if login is implemented).
     - `subscriptions` → Stores user emails who subscribed to the newsletter.
     - `playlists` → Stores curated playlists featured in the system.

4. **Redirection**:
   - **After an action is performed**, the user is redirected or shown a confirmation message.
     - **Subscription:** Redirects to a "Thank You for Subscribing" popup.
     - **Song Selection:** Updates the music player UI dynamically.
     - **Filtering by Genre/Artist:** Updates song lists based on user selection.

Process Logic

1. Front-End Module (User Interaction)

- The user interacts with the interface (HTML, CSS, JavaScript).
- Users can:
  - Browse and play songs.
  - Filter music by genre or artist.
  - Subscribe to the newsletter.
  - Submit an inquiry or feedback.
- If subscribing or submitting a form, JavaScript validation is applied.
- Data is sent to the back-end via a **POST request**

Process Flow (Front-End)

- User → Browse Music & Select → Click Play → Music Player Loads & Plays
- User → Enter Email for Subscription → Click Subscribe → Data Sent to Server
- User → Submit Inquiry Form → JavaScript Validation → Send to PHP

2. Back-End Module (PHP Processing)

- The **PHP script** processes user actions:
  - If it's a music request → Fetch the song from the database.
  - If it's a subscription request → Validate email & store in the database.
  - If it's an inquiry form → Store in the database.
- It ensures **data validation** and handles errors.

Process Flow (Back-End)

PHP Receives Request → Check & Validate Data →If Valid → Store in Database

↓

If Invalid → Show Error Message

3. Database Module (Storage & Retrieval)

- All user interactions (songs played, subscriptions, inquiries) are stored in a **MySQL database**.
- The database has different tables:
  - users → Stores user details.
  - songs → Stores song metadata.
  - playlists → Stores playlist information.
  - subscriptions → Stores email subscriptions.
  - inquiries → Stores user messages and feedback.

Process Flow (Database)

- User → Browse Music & Select → Click Play → Music Player Loads & Plays
- User → Enter Email for Subscription → Click Subscribe → Data Sent to Server
- User → Submit Inquiry Form → JavaScript Validation → Send to PHP

4. Redirection Module (User Feedback)

- After form submission or subscription:
  - If successful → Redirect to **"Thank You"** page with a success message.
  - If unsuccessful → Show an error message.
- For music playback, the user stays on the same page, and the music player updates dynamically.

Process Flow (Redirection)

- Successful Subscription → Redirect to Thank You Page
- Unsuccessful Subscription → Show Error Message
- Successful Inquiry Submission → Redirect to Confirmation Page
- Song Playback → Update Music Player Without Reloading Page

iii) Data Structures and Tables

Data Structures and Tables are the backbone of the **SoundSphere Music System**. They define how music-related data, user interactions, and playlists are organized, stored, and retrieved efficiently. In this project, a MySQL database is used to manage song metadata, user preferences, and form submissions. This ensures that data is systematically stored, easily accessible, and scalable for future enhancements, providing a seamless music streaming experience.

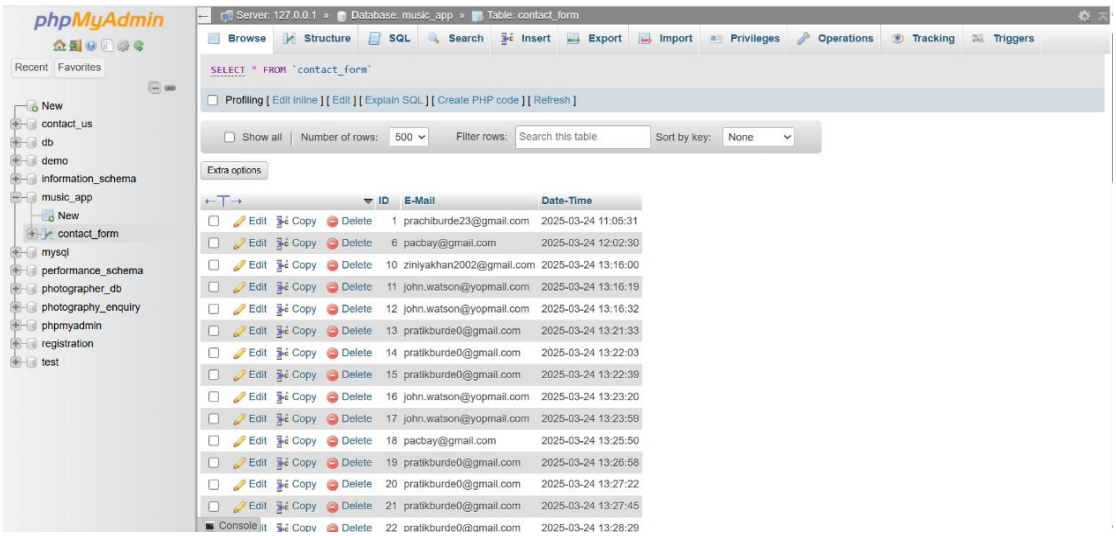


Table Name: contact_form

Field Name	Data Type	Description
ID	INT (AUTO_INCREMENT, PRIMARY KEY)	Unique identifier for each contact form submission.
E-Mail	VARCHAR (255)	Stores the email address of the user making the inquiry.
Date-Time	TIMESTAMP (DEFAULT CURRENT_TIMESTAMP)	Records the date and time when the form was submitted.

iv) Entity-Relationship Diagram

In **SoundSphere**, the system involves user interactions through music browsing, playlist management, and form submissions.

Relationships in Subscription Module

1. **Users & Subscription** → One user can have **one or multiple subscriptions** (if re-subscribed after expiration).

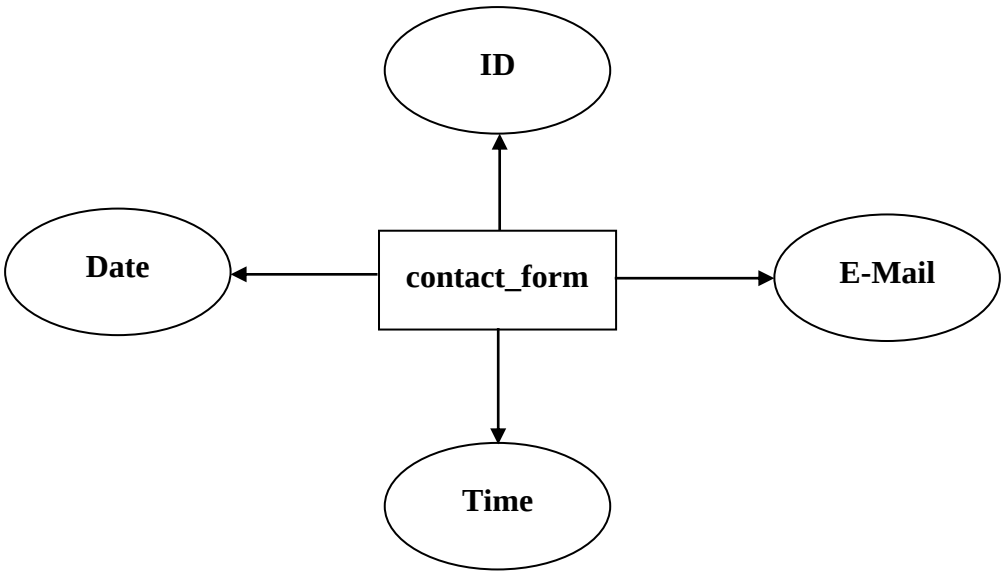
2. **Subscription & Payments (if applicable)** → Each subscription may be linked to **a payment record** (if a premium plan exists).

Entities & Relationships

1. **Users** → {ID, Name, Email, Password}
    - **One-to-One Relationship** with Subscription (Each email can have only one active subscription).

2. **Subscription** → {ID, Email, Time-Date}
    - **Primary Key**: ID
    - **Foreign Key**: Email (references Users table)

3. **Payments (if included)** → {PaymentID, SubscriptionID, Amount, Status}
    - **One-to-One Relationship** with Subscription



7. System Design

Architectural Diagram

- **Client Side (Frontend):**
  - HTML, CSS, JavaScript (React.js or Vanilla JS)
  - UI for browsing, searching, and playing music

- **Server Side (Backend):**
  - PHP or Node.js for handling requests
  - API integration for streaming music

- **Database:**
  - MySQL / PostgreSQL to store users, songs, playlists, and subscriptions

- **Data Flow:**
  - User requests → Server processes → Fetches data from DB → Displays results

i) Page Design

This section covers the layout and design of key pages in the SoundSphere platform.

- **Home Page (home.html)**
  - **Hero section**: Large banner with featured songs
  - **Genre filters**: Buttons to browse different genres
  - **Popular playlists**: Display top playlists in a card format

- **Featured Page (#featured)**
  - **Trending songs**: Popup modal for each song
  - **Categories**: Handpicked best tracks
  - **Search functionality**: Live search bar

- **Retro Tracks Page (retro.html)**
  - **Decade-based filtering** (60s, 70s, 80s, etc.)
  - **Music cards**: Popup play options

- **Artists Page (artists.html)**
  - Artist profile pages
  - List of songs per artist

ii) Source Code

HTML (Home Page)

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Home | SoundSphere </title>
   <link rel="stylesheet" href="home.css">
</head>
<body>
   <header>
      <div class="logo">
         <a href="home.html">SoundSphere </a>
      </div>
      <nav>
         <ul class="nav-links">
            <li><a href="home.html" class="active">Home</a></li>
            <li><a href="#featured" class="active1">Featured</a></li>
            <li><a href="retro.html">Retro Tracks</a></li>
            <li><a href="artists.html">Artists</a></li>
         </ul>
      </nav>
      <div class="search-container">
         <input type="text" class="search-input" placeholder="Search music...">
         <button class="search-btn">
            <i class="fas fa-search"></i>
         </button>
      </div>
      <div class="search-results" id="searchResults">
         <!-- Search results will be populated here -->
      </div>
   </header>
   <!-- Hero Section -->
   <section class="hero" id="hero">
      <div class="hero-content">
         <h1>Discover Your Sound</h1>
         <p>Stream millions of songs, create playlists, and share your favorite tracks
with friends.</p>
         <button id="playButton" class="cta-btn">Start Listening</button>
         <audio id="audioPlayer"></audio>
```

```
        </div>
    </section>
    <!-- Featured Music -->
    <section id="featured">
        <h2 class="sections-title">Featured This Week</h2>
        <div class="music-grid">
            <!-- Album 1 -->
            <div class="music-card" data-collection="60s">
                <div class="music-info">
                    <div class="music-title">60's</div>
                </div>
            </div>
            <div class="music-card" data-collection="70s">
                <div class="music-info">
                    <div class="music-title">70's</div>
                </div>
            </div>
            <div class="music-card" data-collection="80s">
                <div class="music-info">
                    <div class="music-title">80's </div>
                </div>
            </div>
            <div class="music-card" data-collection="90s">
                <div class="music-info">
                    <div class="music-title">90's</div>
                </div>
            </div>
            <div class="music-card" data-collection="2000s">
                <div class="music-info">
                    <div class="music-title">2000's</div>
                </div>
            </div>
        </div>
    </section>
    <div id="musicDetails" class="overlay">
        <div class="popup">
            <span class="exit-icon"><i class="fas fa-times"></i></span>
            <div class="popup-banner">
                <img id="albumCover" class="cover-art" src="images/playlists/90s-
dance.jpg" alt="Playlist">
                <div class="banner-text">
                    <h2 id="collectionName" class="collection-heading">Playlist Title</h2>
                    <p id="collectionSummary" class="collection-summary">Playlist
description goes here</p>
```

```
            </div>
          </div>
          <ul id="trackListing" class="track-container">
            <!-- Tracks will be added here dynamically -->
          </ul>
        </div>
      </div>
      <section class="categories">
        <h2 class="section-title">Browse by Genre</h2>
        <!-- Added filter controls -->
        <div class="filter-controls">
          <button class="filter-btn active" data-filter="all">All Genres</button>
          <button class="filter-btn" data-filter="moods">Moods</button>
          <button class="filter-btn" data-filter="genres">Musical Genres</button>
        </div>
        <div class="category-container">
          <div class="category-card" data-category="moods" data-genre="Romantic"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Romantic</span>
          </div>
          <div class="category-card" data-category="genres" data-genre="Classical"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Classical</span>
          </div>
          <div class="category-card" data-category="moods" data-genre="Sad"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Sad</span>
          </div>
          <div class="category-card" data-category="moods" data-genre="Upbeat"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Upbeat</span>
          </div>
          <div class="category-card" data-category="moods" data-genre="Funny"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Funny</span>
          </div>
          <div class="category-card" data-category="genres" data-genre="Jazz"
            style="background-image: url('/api/placeholder/300/100')">
            <span>Jazz</span>
          </div>
        </div>
        <div class="song-popup" id="songPopup">
          <div class="popup-content">
            <div class="popup-header">
```

```
            <h3 id="popupTitle">Genre Songs</h3>
            <span class="close-btn">&times;</span>
          </div>
          <div class="popup-body">
            <ul id="songList">
              <!-- Songs will be dynamically added here -->
            </ul>
          </div>
        </div>
      </div>
    </section>
    <div class="music-player">
      <div class="now-playing">
     <img src="images/default-album.jpg" alt="Album Cover" id="current-album-art">
        <div class="track-info">
          <div id="current-song">Select a song</div>
          <div id="current-artist">to start playing</div>
        </div>
      </div>
      <div class="player-controls">
        <button id="prev-btn"><i class="fas fa-step-backward"></i></button>
        <button id="play-btn"><i class="fas fa-play"></i></button>
        <button id="next-btn"><i class="fas fa-step-forward"></i></button>
      </div>
      <div class="progress-container">
        <span id="current-time">0:00</span>
        <div class="progress-bar">
          <div class="progress"></div>
        </div>
        <span id="duration">0:00</span>
      </div>
      <div class="volume-container">
        <i class="fas fa-volume-up" id="volumeIcon"></i>
        <div class="volume-slider" id="volumeSlider">
          <div class="volume-progress" id="volumeProgress"></div>
        </div>
      </div>
      <audio id="audioPlayer" src="your-audio-file.mp3"></audio> <!-- Optional -->
    </div>
    <footer>
      <div class="footer-content">
        <div class="footer-column">
          <h3>Company</h3>
          <ul class="footer-links">
```

```
                <li><a href="#">About Us</a></li>
                <li><a href="#">Careers</a></li>
                <li><a href="#">Press</a></li>
                <li><a href="#">News</a></li>
            </ul>
        </div>
        <div class="footer-column">
            <h3>Communities</h3>
            <ul class="footer-links">
                <li><a href="#">For Artists</a></li>
                <li><a href="#">Developers</a></li>
                <li><a href="#">Advertising</a></li>
                <li><a href="#">Investors</a></li>
            </ul>
        </div>
        <div class="footer-column">
            <h3>Useful Links</h3>
            <ul class="footer-links">
                <li><a href="#">Support</a></li>
                <li><a href="#">Web Player</a></li>
                <li><a href="#">Mobile App</a></li>
            </ul>
        </div>
        <div class="footer-column">
            <h3>Connect with Us</h3>
            <p>Stay engaged with our community on social media and never miss an
update.</p>
            <div class="social-links">
                <a href="#"><i class="fab fa-facebook-f"></i></a>
                <a href="#"><i class="fab fa-twitter"></i></a>
                <a href="#"><i class="fab fa-instagram"></i></a>
                <a href="#"><i class="fab fa-youtube"></i></a>
            </div>
        </div>
    </div>
    <div class="copyright">
        <p>&copy; 2025 Rhythm. All rights reserved.</p>
    </div>
</footer>
<script src="home.js"></script>
<script src="https://kit.fontawesome.com/9e5a623fa6.js"
crossorigin="anonymous"></script>
</body>
</html>
```

CSS (Home Page)

```
/* Base Styles */
:root {
    --primary-color: #8a18a4;
    --dark-bg: #121212;
    --darker-bg: #0a0a0a;
    --light-bg: #282828;
    --text-color: #ffffff;
    --secondary-text: #b3b3b3;
    --hover-color: #c437e4;
    --border-color: #333333;
}
/* Header Styles */
header {
    background-color: var(--darker-bg);
    padding: 20px 40px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    position: sticky;
    top: 0;
    z-index: 100;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.3);
}
.logo a {
    color: var(--primary-color);
    font-size: 1.8rem;
    font-weight: bold;
    letter-spacing: 1px;
}
.nav-links {
    display: flex;
    list-style: none;
}
.nav-links li {
    margin: 0 15px;
}
.nav-links a {
    font-weight: 500;
    padding: 5px 0;
    position: relative;
    transition: color 0.3s ease;
```

```
}
.nav-links a::after {
    content: '';
    position: absolute;
    bottom: 0;
    left: 0;
    width: 0;
    height: 2px;
    background-color: var(--primary-color);
    transition: width 0.3s ease;
}
.nav-links a:hover::after,
.nav-links a.active::after {
    width: 100%;
}
.nav-links a.active {
    color: var(--primary-color);
}
a {
    text-decoration: none;
    color: var(--text-color);
    transition: color 0.3s ease;
}
a:hover {
    color: var(--primary-color);
}
button {
    cursor: pointer;
}
/* Base styles for the search container */
.search-container {
    display: flex;
    align-items: center;
    background: #282828;
    padding: 5px 10px;
    border-radius: 20px;
    width: fit-content;
    position: relative;
}
.search-input {
    padding: 8px 12px;
    border: none;
    border-radius: 20px;
    background-color: transparent;
```

```
    color: #ffffff;
    width: 200px;
    outline: none;
}
.search-btn {
    background: #8a18a4;
    border: none;
    border-radius: 50%;
    width: 32px;
    height: 32px;
    cursor: pointer;
    color: white;
    display: flex;
    align-items: center;
    justify-content: center;
}
.search-btn i {
    color: black;
    font-size: 16px;
}
/* Search results styles */
.search-results {
    position: absolute;
    top: 70px;
    right: 25px;
    width: 300px;
    /* Fixed width for desktop */
    background: #282828;
    margin-top: 8px;
    border-radius: 8px;
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.3);
    z-index: 1000;
    max-height: 400px;
    overflow-y: auto;
    display: none;
}
.result-item {
    display: flex;
    padding: 12px;
    align-items: center;
    border-bottom: 1px solid #333;
    cursor: pointer;
    transition: background 0.2s;
}
```

```
.result-item:hover {
    background: #333;
}
.result-image {
    width: 40px;
    height: 40px;
    border-radius: 4px;
    margin-right: 12px;
    object-fit: cover;
}
.result-info {
    flex: 1;
}
.result-title {
    color: #fff;
    font-weight: 500;
    margin-bottom: 4px;
}
.result-artist {
    color: #b3b3b3;
    font-size: 12px;
}
.result-type {
    font-size: 11px;
    text-transform: uppercase;
    color: #b3b3b3;
    padding: 2px 8px;
    border-radius: 10px;
    background: #333;
    margin-left: auto;
}
.result-section-title {
    padding: 8px 12px;
    color: #b3b3b3;
    font-size: 14px;
    font-weight: 600;
    background: #222;
}
.no-results {
    padding: 16px;
    text-align: center;
    color: #b3b3b3;
}
/* Hero Section */
```

```
.hero {
    max-width: 1300px;
    margin: 0 auto;
    padding: 0 20px;
    background-image: linear-gradient(rgba(0, 0, 0, 0.6), rgba(0, 0, 0, 0.4)),
url('abstract.jpg');
    background-size: cover;
    background-position: center;
    height: 620px;
    display: flex;
    align-items: center;
    justify-content: center;
    text-align: center;
    color: var(--text-color);
    margin-bottom: 20px;
    border-radius: 10px;
}
.hero::before {
    content: "";
    position: absolute;
    inset: 0;
}
.hero-content {
    position: relative;
    z-index: 2;
    max-width: 600px;
    text-align: left;
    margin-left: -500px;
}
.hero h1 {
    font-size: 3rem;
    margin-bottom: 20px;
    opacity: 0.9;
}
.hero p {
    font-size: 1.2rem;
    margin-bottom: 30px;
    max-width: 700px;
    opacity: 0.9;
}
.cta-btn {
    background: #8a18a4;
    color: white;
    padding: 12px 18px;
```

```
    font-size: 16px;
    font-weight: 600;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: 0.3s;
}
.cta-btn:hover {
    background-color: #c437e4;
}
/* Featured Section */
#featured {
    padding: 60px 5%;
    text-align: center;
}
.sections-title {
    font-size: 2rem;
    margin-bottom: 40px;
    position: relative;
    padding-bottom: 10px;
}
.sections-titles::after {
    content: '';
    position: absolute;
    bottom: 0;
    left: 50%;
    transform: translateX(-50%);
    width: 60px;
    height: 3px;
    background-color: #8a18a4;
}
.music-grid {
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 50px;
    flex-wrap: wrap;
}
.music-card {
    background: linear-gradient(135deg, #8a18a4, #2e013a);
    border-radius: 8px;
    overflow: hidden;
    transition: transform 0.3s, background 0.5s;
    cursor: pointer;
```

```
    height: 170px;
    width: 200px;
    display: flex;
    align-items: center;
    justify-content: center;
    text-align: center;
    position: relative;
}
.music-card:hover {
    transform: translateY(-10px);
    background: linear-gradient(135deg, #2e013a, #8a18a4);
}
.music-info {
    color: white;
    font-weight: bold;
}
.music-title {
    font-weight: bold;
    margin-bottom: 5px;
    font-size: 1.4rem;
}
/* Modal Overlay Styles */
.overlay {
    display: none;
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.8);
    z-index: 1000;
    align-items: center;
    justify-content: center;
}
.popup {
    background-color: #1e1e1e;
    max-width: 500px;
    border-radius: 10px;
    overflow: hidden;
    box-shadow: 0 0 20px rgba(0, 0, 0, 0.6);
    position: relative;
    animation: popupEntry 0.3s ease-out;
}
.popup-banner {
```

```
    display: flex;
    padding: 20px;
    background: linear-gradient(135deg, #8a18a4, #2e013a);
}
.cover-art {
    width: 150px;
    height: 150px;
    border-radius: 8px;
    object-fit: cover;
    margin-right: 20px;
}
.banner-text {
    display: flex;
    flex-direction: column;
    justify-content: center;
}
.collection-heading {
    font-size: 1.8rem;
    margin-bottom: 8px;
}
.collection-summary {
    font-size: 1rem;
    opacity: 0.8;
}
.exit-icon {
    position: absolute;
    top: 15px;
    right: 15px;
    font-size: 1.5rem;
    color: white;
    cursor: pointer;
    transition: color 0.2s;
}
.exit-icon:hover {
    color: #000000;
}
.track-container {
    list-style: none;
    padding: 0;
}
.track-item {
    display: flex;
    align-items: center;
    padding: 12px 20px;
```

```
    transition: background-color 0.2s;
    border-bottom: 1px solid #333;
}
.track-item:hover {
    background-color: #2a2a2a;
}
.track-number {
    width: 30px;
    font-size: 0.9rem;
    opacity: 0.5;
}
.track-control {
    margin-right: 15px;
    cursor: pointer;
    width: 30px;
    height: 30px;
    border-radius: 50%;
    background-color: #333;
    display: flex;
    align-items: center;
    justify-content: center;
    transition: background-color 0.2s;
}
.track-control:hover {
    background-color: #8a18a4;
}
.track-details {
    flex: 1;
}
.track-name {
    font-weight: 500;
    margin-bottom: 3px;
}
.track-artist {
    font-size: 0.9rem;
    opacity: 0.7;
}
.track-length {
    font-size: 0.9rem;
    opacity: 0.7;
}
/* Popup styling */
.song-popup {
    position: fixed;
```

```
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.7);
    display: none;
    justify-content: center;
    align-items: center;
    z-index: 1000;
    opacity: 0;
    transition: opacity 0.3s ease;
}
.song-popup.active {
    display: flex;
    opacity: 1;
}
.popup-content {
    background-color: black;
    border-radius: 10px;
    max-width: 500px;
    max-height: 80vh;
    overflow: hidden;
    box-shadow: 0 5px 30px rgba(0, 0, 0, 0.3);
    animation: scaleUp 0.3s ease forwards;
}
.popup-header {
    padding: 20px;
    background: linear-gradient(135deg, #8a18a4, #2e013a);
    color: white;
    display: flex;
    justify-content: space-between;
    align-items: center;
}
.popup-header h3 {
    margin: 0;
    font-size: 1.5rem;
}
.close-btn {
    font-size: 2rem;
    cursor: pointer;
    transition: transform 0.2s ease;
}
.close-btn:hover {
    transform: scale(1.2);
```

```
}
.popup-body {
    padding: 20px;
    max-height: calc(80vh - 80px);
    overflow-y: auto;
}
#songList {
    list-style: none;
    padding: 0;
    margin: 0;
}
#songList li {
    padding: 12px 15px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    transition: background-color 0.2s ease;
}
#songList li:last-child {
    border-bottom: none;
}
#songList li:hover {
    background-color: #464646;
}
#songList li .song-title {
    font-weight: bold;
}
#songList li .song-artist {
    color: #777;
    font-size: 0.9rem;
}
#songList li .play-icon {
    color: #8a18a4;
    cursor: pointer;
    opacity: 0.7;
    transition: opacity 0.2s ease;
}
#songList li .play-icon:hover {
    opacity: 1;
}
/* Base styles */
.section-title {
    text-align: center;
    font-size: 2.5rem;
```

```
    margin-bottom: 30px;
    color: white;
    position: relative;
}
.section-title::after {
    content: '';
    position: absolute;
    bottom: -10px;
    left: 50%;
    transform: translateX(-50%);
    width: 50px;
    height: 3px;
    background-color: #8a18a4;
}
.category-container {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));
    gap: 20px;
    max-width: 1200px;
    margin: 0 auto;
    padding-top: 20px;
    padding-left: 10px;
    padding-right: 10px;
}
.category-card {
    height: 180px;
    border-radius: 10px;
    overflow: hidden;
    position: relative;
    cursor: pointer;
    background-size: cover;
    background-position: center;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
    transition: all 0.3s ease;
}
.category-card span {
    position: absolute;
    bottom: 0;
    left: 0;
    width: 100%;
    padding: 15px;
    color: white;
    font-size: 1.2rem;
    font-weight: bold;
```

```
    background: linear-gradient(to top, rgba(0, 0, 0, 0.7), transparent);
    transform: translateY(0);
    transition: all 0.3s ease;
}
.category-card:hover {
    transform: translateY(-10px) scale(1.03);
    box-shadow: 0 15px 30px rgba(0, 0, 0, 0.2);
}
.category-card:hover span {
    padding-bottom: 25px;
    background: linear-gradient(to top, rgba(149, 25, 160, 0.8), transparent);
}
.category-card.active {
    border: 3px solid #8a18a4;
    transform: translateY(-5px) scale(1.02);
}
.filter-controls {
    display: flex;
    gap: 10px;
    justify-content: center;
    margin-bottom: 25px;
    flex-wrap: wrap;
}
.filter-btn {
    background-color: #fff;
    border: 2px solid #ddd;
    border-radius: 25px;
    padding: 8px 20px;
    cursor: pointer;
    font-weight: bold;
    transition: all 0.2s ease;
}
.filter-btn:hover {
    border-color: #c437e4;
    color: #c437e4;
}
.filter-btn.active {
    background-color: #c437e4;
    color: white;
    border-color: #c437e4;
}
/* Popup styling */
.song-popup {
    position: fixed;
```

```
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.7);
    display: none;
    justify-content: center;
    align-items: center;
    z-index: 1000;
    opacity: 0;
    transition: opacity 0.3s ease;
}
.song-popup.active {
    display: flex;
    opacity: 1;
}
.popup-content {
    background-color: black;
    border-radius: 10px;
    width: 90%;
    max-width: 500px;
    max-height: 80vh;
    overflow: hidden;
    box-shadow: 0 5px 30px rgba(0, 0, 0, 0.3);
    animation: scaleUp 0.3s ease forwards;
}
.popup-header {
    padding: 20px;
    background-color: #8a18a4;
    display: flex;
    justify-content: space-between;
    align-items: center;
}
.popup-header h3 {
    margin: 0;
    font-size: 1.5rem;
}
.close-btn {
    font-size: 2rem;
    cursor: pointer;
    transition: transform 0.2s ease;
}
.close-btn:hover {
    transform: scale(1.2);
```

```
}
.popup-body {
    padding: 20px;
    max-height: calc(80vh - 80px);
    overflow-y: auto;
}
#songList {
    list-style: none;
    padding: 0;
    margin: 0;
}
#songList li {
    padding: 12px 15px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    transition: background-color 0.2s ease;
}
#songList li:last-child {
    border-bottom: none;
}
#songList li:hover {
    background-color: #292727;
}
#songList li .song-title {
    font-weight: bold;
}
#songList li .song-artist {
    color: #777;
    font-size: 0.9rem;
}
#songList li .play-icon {
    color: #8a18a4;
    cursor: pointer;
    opacity: 0.7;
    transition: opacity 0.2s ease;
}
#songList li .play-icon:hover {
    opacity: 1;
}
    #songList li {
        padding: 10px 12px;
        flex-direction: column;
        align-items: flex-start;
```

```
    }
    #songList li .song-title {
        margin-bottom: 5px;
    }
    #songList li .song-artist {
        margin-bottom: 10px;
    }
    #songList li .play-icon {
        align-self: flex-end;
    }
}
/* Music Player */
.music-player {
    position: fixed;
    bottom: 0;
    left: 0;
    right: 0;
    background-color: var(--darker-bg);
    padding: 15px 40px;
    display: flex;
    align-items: center;
    justify-content: space-between;
    border-top: 1px solid var(--border-color);
    z-index: 100;
}
.now-playing {
    display: flex;
    align-items: center;
    width: 25%;
}
.now-playing img {
    width: 56px;
    height: 56px;
    border-radius: 4px;
    margin-right: 15px;
}
.track-info {
    display: flex;
    flex-direction: column;
}
#current-song {
    font-weight: 500;
    margin-bottom: 5px;
}
```

```
#current-artist {
    color: var(--secondary-text);
    font-size: 0.85rem;
}
.player-controls {
    display: flex;
    align-items: center;
    gap: 20px;
}
.player-controls button {
    background: transparent;
    border: none;
    color: var(--text-color);
    font-size: 16px;
    cursor: pointer;
    transition: transform 0.2s, color 0.2s;
}
.player-controls button:hover {
    color: var(--primary-color);
    transform: scale(1.1);
}
#play-btn {
    width: 40px;
    height: 40px;
    border-radius: 50%;
    background: var(--text-color);
    color: var(--darker-bg);
    font-size: 18px;
    display: flex;
    align-items: center;
    justify-content: center;
}
#play-btn:hover {
    background: var(--primary-color);
    color: black;
}
.progress-container {
    flex: 1;
    display: flex;
    align-items: center;
    gap: 10px;
    padding: 0 30px;
}
#current-time,
```

```
#duration {
    color: var(--secondary-text);
    font-size: 0.8rem;
    width: 40px;
}
.progress-bar {
    flex: 1;
    height: 4px;
    background-color: #555;
    border-radius: 2px;
    cursor: pointer;
    position: relative;
}
.progress {
    height: 100%;
    background-color: var(--primary-color);
    border-radius: 2px;
    width: 0%;
    position: relative;
}
.progress-bar:hover .progress::after {
    opacity: 1;
}
.volume-container {
    display: flex;
    align-items: center;
    gap: 10px;
    width: 200px;
    background: #000;
    padding: 10px;
}
.volume-container i {
    color: white;
    font-size: 20px;
    cursor: pointer;
    transition: color 0.3s;
}
.volume-slider {
    flex: 1;
    height: 5px;
    background: #666;
    border-radius: 2px;
    position: relative;
    cursor: pointer;
```

```
}
.volume-progress {
    height: 100%;
    width: 50%;
    background: #ccc;
    border-radius: 2px;
    transition: width 0.2s ease-in-out;
}
.music-player .close-btn {
    position: absolute;
    top: 10px;
    right: 10px;
    background: none;
    border: none;
    color: white;
    font-size: 24px;
    cursor: pointer;
    z-index: 10;
    transition: color 0.3s ease;
}
.music-player .close-btn:hover {
    color: #8a18a4;
}
* Footer */
footer {
    background-color: #0c0c0c;
    padding: 40px 5% 80px;
    margin-top: 60px;
}
.footer-content {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));
    gap: 30px;
}
.footer-column h3 {
    font-size: 1.2rem;
    margin-bottom: 20px;
    color: #8a18a4;
}
.footer-links {
    list-style: none;
    padding: 0;
}
.footer-links li {
```

```
    margin-bottom: 10px;
}
.footer-links a {
    text-decoration: none;
    color: #b3b3b3;
    transition: color 0.3s;
}
.footer-links a:hover {
    color: #ffffff;
}
.social-links {
    margin-top: 30px;
    margin-bottom: 30px;
    display: flex;
    gap: 30px;
}
.social-links a {
    color: white;
    font-size: 20px;
    transition: 0.3s;
}
.social-links a:hover {
    transform: scale(1.1);
}
.social-links a:hover .fa-facebook-f {
    color: #5078cf;
}
.social-links a:hover .fa-twitter {
    color: #00acee;
}
.social-links a:hover .fa-instagram {
    color: #fd9217;
}
.social-links a:hover .fa-youtube {
    color: #ff0000;
}
.copyright {
    margin-top: 40px;
    text-align: center;
    color: #b3b3b3;
    font-size: 0.9rem;
}
```

JavaScript (Home Page)

```
//SEARCH BAR of ALL PAGE
document.addEventListener('DOMContentLoaded', function () {
    const searchInput = document.querySelector('.search-input');
    const searchBtn = document.querySelector('.search-btn');
    const searchContainer = document.querySelector('.search-container');
    let searchResults = document.getElementById('searchResults');
    if (!searchResults) {
        searchResults = document.createElement('div');
        searchResults.className = 'search-results';
        searchResults.id = 'searchResults';
                                    searchContainer.parentNode.insertBefore(searchResults,
searchContainer.nextSibling);
    }
    let searchTimeout = null;
    function handleSearchInput() {
        const query = searchInput.value.trim();
        if (searchTimeout) {
            clearTimeout(searchTimeout);
        }
        if (query.length === 0) {
            searchResults.style.display = 'none';
            return;
        }
        searchResults.style.display = 'block';
        searchResults.innerHTML = '<div class="no-results">Searching...</div>';
        searchTimeout = setTimeout(() => {
            performSearch(query);
        }, 300);
    }
    function performSearch(query) {
        setTimeout(() => {
            const results = fetchMockResults(query);
            displayResults(results, query);
        }, 500);
    }
    function fetchMockResults(query) {
        const allItems = [
            { id: 1, title: 'Mera Joota Hai Japani', artist: 'Mukesh (Shree 420)', type: 'song',
image: 'https://via.placeholder.com/40' },
            { id: 2, title: 'Pyar Kiya To Darna Kya', artist: 'Lata Mangeshkar (Mughal-e-
Azam)', type: 'album', image: 'https://via.placeholder.com/40' },
            { id: 3, title: 'Aye Meri Zohra Jabeen', artist: 'Manna Dey (Waqt)', type: 'artist',
image: 'https://via.placeholder.com/40' },
            { id: 4, title: 'Aaj Kal Tere Mere Pyaar Ke Charche', artist: 'Mohammed Rafi
(Brahmachari)', type: 'song', image: 'https://via.placeholder.com/40' },
            //Many more
        ];
```

```
        return allItems.filter(item => {
            const titleMatch = item.title.toLowerCase().includes(query.toLowerCase());
            const artistMatch = item.artist.toLowerCase().includes(query.toLowerCase());
            return titleMatch || artistMatch;
        });
    }
    function displayResults(results, query) {
        searchResults.innerHTML = '';
        if (results.length === 0) {
            searchResults.innerHTML = `<div class="no-results">No results found for "${query}"</div>`;
            return;
        }
        const groupedResults = {
            artist: [],
            album: [],
            song: [],
            playlist: []
        };
        results.forEach(item => {
            if (groupedResults[item.type]) {
                groupedResults[item.type].push(item);
            }
        });
        const types = ['artist', 'album', 'song', 'playlist'];
        types.forEach(type => {
            const typeItems = groupedResults[type];
            if (typeItems.length > 0) {
                const sectionTitle = document.createElement('div');
                sectionTitle.className = 'result-section-title';
                sectionTitle.innerHTML = `<div style="padding: 8px 12px; color: #b3b3b3; font-size: 14px; font-weight: 600;">${type.charAt(0).toUpperCase() + type.slice(1)}s</div>`;
                searchResults.appendChild(sectionTitle);

                typeItems.slice(0, 3).forEach(item => { // Limit to 3 items per section like Spotify
                    const resultItem = createResultItem(item);
                    searchResults.appendChild(resultItem);
                });
            }
        });
        searchResults.style.display = 'block';
    }
    function createResultItem(item) {
        const resultItem = document.createElement('div');
        resultItem.className = 'result-item';
        resultItem.innerHTML = `
            <img src="${item.image}" alt="" class="result-image">
            <div class="result-info">
```

```
            <div class="result-title">${item.title}</div>
            ${item.artist ? `<div class="result-artist">${item.artist}</div>` : ''}
          </div>
          <div class="result-type">${item.type}</div>
        `;
    resultItem.addEventListener('click', () => {
        console.log(`Clicked on ${item.type}: ${item.title}`);
        if (item.type === 'song') {
            alert(`Now playing: ${item.title} by ${item.artist}`);
        } else {
            alert(`Opening ${item.type}: ${item.title}`);
        }
        searchResults.style.display = 'none';
        searchInput.value = item.title;
    });
    return resultItem;
}

function handleSearchButtonClick() {
    if (searchInput.value.trim().length > 0) {
        performSearch(searchInput.value.trim());
    }
}
searchInput.addEventListener('input', handleSearchInput);
searchInput.addEventListener('focus', function () {
    if (searchInput.value.trim().length > 0) {
        searchResults.style.display = 'block';
    }
});
searchBtn.addEventListener('click', handleSearchButtonClick);
document.addEventListener('click', function (event) {
    if (!searchContainer.contains(event.target) &&
        !searchResults.contains(event.target)) {
        searchResults.style.display = 'none';
    }
});

const ctaButton = document.querySelector(".cta-btn");

const musicCards = document.querySelectorAll('.music-card');
musicCards.forEach(card => {
    card.addEventListener('click', function () {
        const musicTitle = this.querySelector('.music-title').textContent;
        const musicArtist = this.querySelector('.music-artist').textContent;
        const albumCover = this.querySelector('.music-thumbnail').src;
        document.querySelector('.track-name').textContent = musicTitle;
        document.querySelector('.artist-name').textContent = musicArtist;
        document.querySelector('.now-playing img').src = albumCover;
        document.querySelector('.progress').style.width = '0%';
        document.querySelector('#play-btn').textContent = '⏸';
```

```
            simulatePlayback();
        });
    });
});

// FIRST MUSIC SECTION OR MUSIC BUTTON
document.addEventListener('DOMContentLoaded', function () {
    // Get DOM elements
    const searchInput = document.querySelector('.search-input');
    const searchBtn = document.querySelector('.search-btn');
    const searchResults = document.getElementById('searchResults');
    const playButton = document.getElementById('playButton');
    const audioPlayer = document.getElementById('audioPlayer');
    const playBtn = document.getElementById('play-btn');
    const prevBtn = document.getElementById('prev-btn');
    const nextBtn = document.getElementById('next-btn');
    const currentSongDisplay = document.getElementById('current-song');
    const currentArtistDisplay = document.getElementById('current-artist');
    const progressElement = document.querySelector('.progress');
    const currentTimeDisplay = document.getElementById('current-time');
    const durationDisplay = document.getElementById('duration');
    const albumArt = document.getElementById('current-album-art');
    const songs = [
        { title: "Aap Ke Aa Jane Se", artist: "Mohammed Aziz & Sadhana Sargam", src:
"90s/Aap Ke Aa Jane Se.mp3", cover: "images/apke.jpg" },
        { title: "Aankh Hai Bhari Bhari", artist: "Kumar Sanu", src: "90s/Aankh Hai
Bhari Bhari (Male).mp3", cover: "images/aankh.jpg" },
        { title: "Aisi Deewangi", artist: "Alka Yagnik, Nadeem–Shravan, and Vinod
Rathod", src: "90s/Aisi Deewangi.mp3", cover: "images/aisi.jpg" },
        //Many more
    ];
    let currentSongIndex = 0;
    let isPlaying = false;
    function playSong(song) {
        audioPlayer.src = song.src;
        currentSongDisplay.textContent = song.title;
        currentArtistDisplay.textContent = song.artist;
        if (albumArt) {
            console.log("Setting album cover to:", song.cover);
            albumArt.src = song.cover;
        } else {
            console.error("Album art element not found");
        }
        audioPlayer.play().then(() => {
            isPlaying = true;
            updatePlayButton();
        }).catch(error => {
            console.error("Error playing audio:", error);
        });
    }
```

```
function handleSearchInput() {
    const query = searchInput.value.trim().toLowerCase();
    if (query.length === 0) {
        searchResults.innerHTML = '';
        return;
    }
    const results = songs.filter(song =>
        song.title.toLowerCase().includes(query) ||
        song.artist.toLowerCase().includes(query)
    );
    displaySearchResults(results);
}
function displaySearchResults(results) {
    searchResults.innerHTML = '';
    if (results.length === 0) {
        searchResults.innerHTML = '<div class="no-results">No results found</div>';
        return;
    }
    results.forEach(song => {
        const resultItem = document.createElement('div');
        resultItem.className = 'result-item';
        resultItem.innerHTML = `<div>${song.title} - ${song.artist}</div>`;
        resultItem.addEventListener('click', () => {
            // Find the index of the selected song
            currentSongIndex = songs.findIndex(s => s.title === song.title);
            playSong(song);
        });
        searchResults.appendChild(resultItem);
    });
}

function pauseSong() {
    audioPlayer.pause();
    isPlaying = false;
    updatePlayButton();
}
function resumeSong() {
    audioPlayer.play();
    isPlaying = true;
    updatePlayButton();
}
function updatePlayButton() {
    // Update both the main play button and the player control button
    if (isPlaying) {
        playButton.textContent = 'Pause';
        playBtn.innerHTML = '<i class="fas fa-pause"></i>';
    } else {
        playButton.textContent = 'Start Listening';
        playBtn.innerHTML = '<i class="fas fa-play"></i>';
    }
```

```
}
playButton.addEventListener('click', function () {
    if (isPlaying) {
        pauseSong();
    } else {
        if (audioPlayer.src) {
            resumeSong();
        } else {
            playSong(songs[currentSongIndex]);
        }
    }
});

playBtn.addEventListener('click', function () {
    if (isPlaying) {
        pauseSong();
    } else {
        if (audioPlayer.src) {
            resumeSong();
        } else {
            playSong(songs[currentSongIndex]);
        }
    }
});

nextBtn.addEventListener('click', function () {
    currentSongIndex = (currentSongIndex + 1) % songs.length;
    playSong(songs[currentSongIndex]);
});
prevBtn.addEventListener('click', function () {
    currentSongIndex = (currentSongIndex - 1 + songs.length) % songs.length;
    playSong(songs[currentSongIndex]);
});
audioPlayer.addEventListener('timeupdate', function () {
    const currentTime = audioPlayer.currentTime;
    const duration = audioPlayer.duration;
    progressElement.style.width = `${(currentTime / duration) * 100}%`;
    currentTimeDisplay.textContent = formatTime(currentTime);
    durationDisplay.textContent = formatTime(duration);
});
function formatTime(seconds) {
    const minutes = Math.floor(seconds / 60);
    const secs = Math.floor(seconds % 60);
    return `${minutes}:${secs < 10 ? '0' : ''}${secs}`;
}
searchInput.addEventListener('input', handleSearchInput);
searchBtn.addEventListener('click', handleSearchInput);
currentSongDisplay.textContent = songs[0].title;
currentArtistDisplay.textContent = songs[0].artist;
audioPlayer.src = songs[0].src;
```

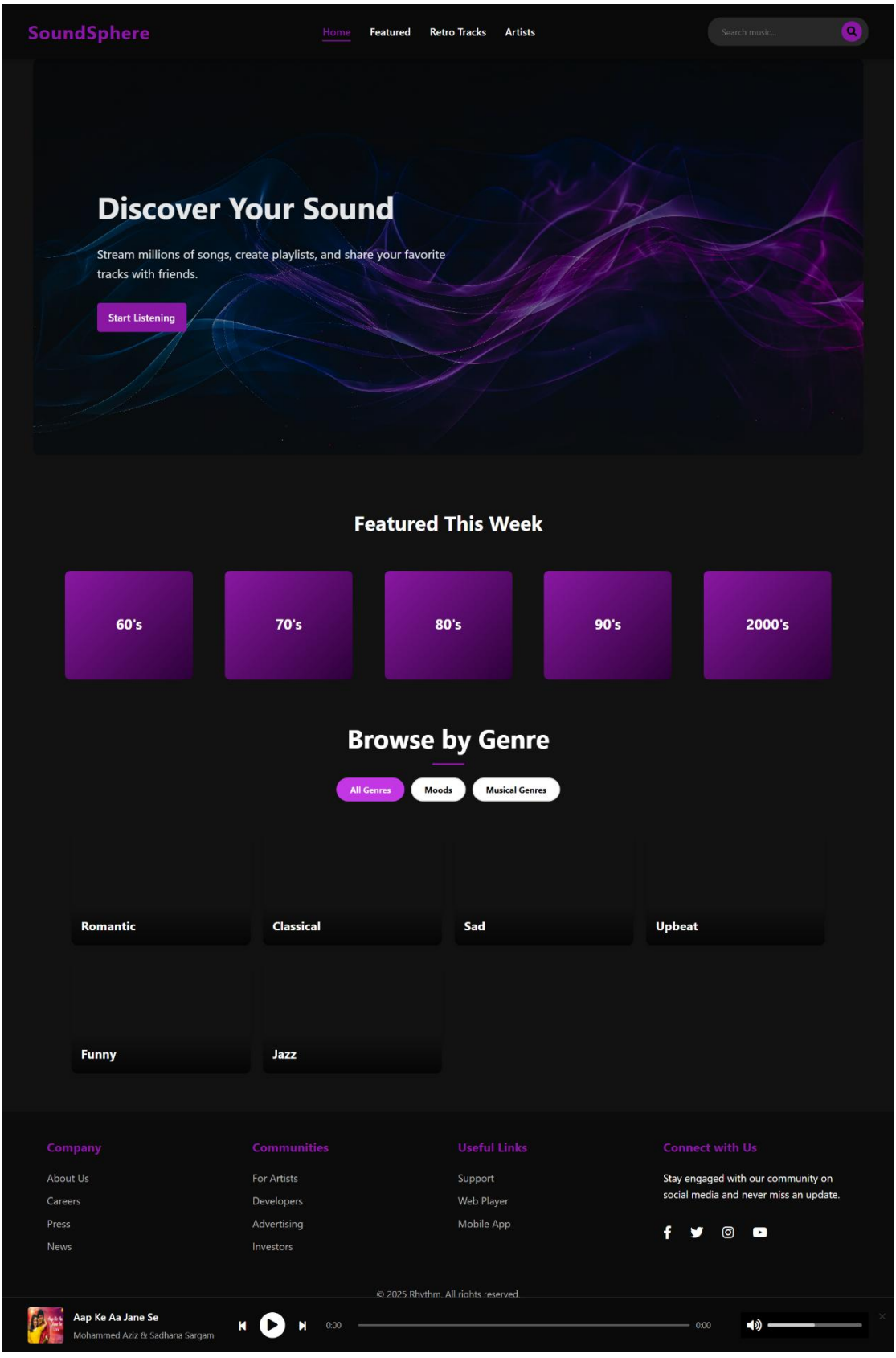
```
    if (albumArt) {
        albumArt.src = songs[0].cover;
    }
    updatePlayButton();
    audioPlayer.addEventListener('ended', function () {
        nextBtn.click();
    });
});

document.addEventListener("DOMContentLoaded", function () {
    const volumeSlider = document.getElementById("volumeSlider");
    const volumeProgress = document.getElementById("volumeProgress");
    const volumeIcon = document.getElementById("volumeIcon");
    const audio = document.getElementById("audioPlayer"); // Optional for real audio
    let volumeLevel = 50; // Default volume level (50%)
    function updateVolume(level) {
        volumeLevel = level;
        volumeProgress.style.width = `${level}%`;
        if (level === 0) {
            volumeIcon.className = "fas fa-volume-mute";
        } else if (level <= 40) {
            volumeIcon.className = "fas fa-volume-down";
        } else {
            volumeIcon.className = "fas fa-volume-up";
        }
        if (audio) {
            audio.volume = level / 100;
        }
    }
    volumeSlider.addEventListener("click", function (e) {
        const rect = volumeSlider.getBoundingClientRect();
        let newVolume = ((e.clientX - rect.left) / rect.width) * 100;
        newVolume = Math.max(0, Math.min(100, newVolume));
        updateVolume(newVolume);
    });

    let isDragging = false;
    volumeSlider.addEventListener("mousedown", function () {
        isDragging = true;
    });
    document.addEventListener("mousemove", function (e) {
        if (isDragging) {
            const rect = volumeSlider.getBoundingClientRect();
            let newVolume = ((e.clientX - rect.left) / rect.width) * 100;
            newVolume = Math.max(0, Math.min(100, newVolume));
            updateVolume(newVolume);
        }
    });

    document.addEventListener("mouseup", function () {
```

```
        isDragging = false;
    });
    volumeIcon.addEventListener("click", function () {
        if (volumeLevel > 0) {
            updateVolume(0); // Mute
        } else {
            updateVolume(50); // Restore default
        }
    });

    const musicPlayer = document.querySelector('.music-player');
    const closeButton = document.createElement('button');
    closeButton.innerHTML = '&times;'; // × symbol
    closeButton.classList.add('close-btn');
    closeButton.setAttribute('title', 'Close Music Player');
    closeButton.style.position = 'absolute';
    closeButton.style.top = '10px';
    closeButton.style.right = '10px';
    closeButton.style.background = 'none';
    closeButton.style.border = 'none';
    closeButton.style.color = '#333';
    closeButton.style.fontSize = '24px';
    closeButton.style.cursor = 'pointer';
    closeButton.style.zIndex = '10';
    musicPlayer.appendChild(closeButton);
    closeButton.addEventListener('click', function() {
        const audioPlayer = document.getElementById('audioPlayer');
        if (audioPlayer) {
            audioPlayer.pause();
        }
        musicPlayer.style.display = 'none';
    });
    updateVolume(volumeLevel);
});

// FEARTUED MUSIC OF HOME
document.addEventListener('DOMContentLoaded', function () {
    const musicDetails = document.getElementById('musicDetails');
    const albumCover = document.getElementById('albumCover');
    const collectionName = document.getElementById('collectionName');
    const collectionSummary = document.getElementById('collectionSummary');
    const trackListing = document.getElementById('trackListing');
    const exitIcon = document.querySelector('.exit-icon');

    const musicCollections = {
        "60s": {
            title: "60's Bollywood Classics",
            description: "Melodious gems from Rafi, Lata & Kishore Kumar",
            image: "images/60's Bollywood Classics.jpg",
            tracks: [
```

```
            { title: "Mera Joota Hai Japani", artist: "Mukesh (Shree 420)", duration: "4:12", path: "" },
            { title: "Pyar Kiya To Darna Kya", artist: "Lata Mangeshkar (Mughal-e-Azam)", duration: "3:43", path: "" },
            { title: "Aye Meri Zohra Jabeen", artist: "Manna Dey (Waqt)", duration: "4:05", path: "" },
            { title: "Aaj Kal Tere Mere Pyaar Ke Charche", artist: "Mohammed Rafi (Brahmachari)", duration: "5:21", path: "" },
            { title: "Mere Mehboob Qayamat Hogi", artist: "Mohammed Rafi (Mr. X in Bombay)", duration: "3:57", path: "" }
          ]
      },

      "70s": {
        title: "70's Bollywood Classics",
        description: "Melodious gems from Rafi, Lata & Kishore Kumar",
        image: "images/70's Bollywood Disco.jpg",
        tracks: [
          { title: "Dum Maro Dum", artist: "Asha Bhosle (Hare Rama Hare Krishna)", duration: "4:35", path: "" },
            { title: "Yamma Yamma", artist: "Kishore Kumar (Shaan)", duration: "5:12", path: "" },
            { title: "Khaike Paan Banaras Wala", artist: "Kishore Kumar (Don)", duration: "4:23", path: "" },
            { title: "Mehbooba Mehbooba", artist: "RD Burman (Sholay)", duration: "3:47", path: "" },
            { title: "Aap Jaisa Koi", artist: "Nazia Hassan (Qurbani)", duration: "4:51", path: "" }
          ]
      },

      "80s": {
        title: "80's Bollywood Classics",
        description: "Melodious gems from Rafi, Lata & Kishore Kumar",
        image: "images/80's Bollywood Dance Hits.jpg",
        tracks: [
            { title: "I Am A Disco Dancer", artist: "Vijay Benedict (Disco Dancer)", duration: "5:23", path: "" },
            { title: "Jimmy Jimmy Jimmy Aaja", artist: "Parvati Khan (Disco Dancer)", duration: "3:55", path: "" },
            { title: "Ek Do Teen", artist: "Alka Yagnik (Tezaab)", duration: "4:42", path: "" },
            { title: "Jumma Chumma De De", artist: "Sudesh Bhosle (Hum)", duration: "6:21", path: "" },
          { title: "Tamma Tamma Loge", artist: "Bappi Lahiri (Thanedaar)", duration: "5:12", path: "" }
          ]
      },

      "90s": {
```

```
            title: "90's Bollywood Classics",
            description: "Melodious gems from Rafi, Lata & Kishore Kumar",
            image: "images/90's romance.jpg",
            tracks: [
                    { title: "Tujhe Dekha To", artist: "Kumar Sanu & Lata Mangeshkar (DDLJ)", duration: "5:02", path: "" },
                { title: "Pehla Nasha", artist: "Udit Narayan & Sadhana Sargam (Jo Jeeta Wohi Sikandar)", duration: "4:49", path: "" },
                    { title: "Ae Mere Humsafar", artist: "Udit Narayan & Alka Yagnik (Baazigar)", duration: "5:37", path: "" },
                { title: "Tu Cheez Badi Hai Mast Mast", artist: "Udit Narayan & Kavita Krishnamurthy (Mohra)", duration: "4:55", path: "" },
                    { title: "Mehndi Laga Ke Rakhna", artist: "Lata Mangeshkar & Udit Narayan (DDLJ)", duration: "4:12", path: "" }
            ]
        },
        "2000s": {
            title: "2000's Bollywood Classics",
            description: "Melodious gems from Rafi, Lata & Kishore Kumar",
            image: "images/2000's Bollywood Fusion.webp",
            tracks: [
                { title: "Kal Ho Naa Ho", artist: "Sonu Nigam (Kal Ho Naa Ho)", duration: "5:21", path: "" },
                    { title: "It's The Time To Disco", artist: "KK, Shaan & Vasundhara Das (Kal Ho Naa Ho)", duration: "5:34", path: "" },
                    { title: "Dhoom Machale", artist: "Sunidhi Chauhan (Dhoom)", duration: "3:56", path: "" },
                { title: "Kajra Re", artist: "Alisha Chinai, Shankar Mahadevan & Javed Ali (Bunty Aur Babli)", duration: "5:48", path: "" },
                    { title: "Mauja Hi Mauja", artist: "Mika Singh (Jab We Met)", duration: "4:03", path: "" }
            ]
        },
    };

    document.querySelectorAll('.music-card').forEach(card => {
        card.addEventListener('click', function () {
            const collectionId = this.getAttribute('data-collection');
            const collection = musicCollections[collectionId];
            if (!collection) return;
            albumCover.src = collection.image;
            albumCover.alt = collection.title;
            collectionName.textContent = collection.title;
            collectionSummary.textContent = collection.description;
            trackListing.innerHTML = '';
            collection.tracks.forEach((track, index) => {
                const li = document.createElement('li');
                li.className = 'track-item';
                li.innerHTML = `
            <div class="track-number">${index + 1}</div>
```

```
            <div class="track-control"><i class="fas fa-play"></i></div>
            <div class="track-details">
                <div class="track-name">${track.title}</div>
                <div class="track-artist">${track.artist}</div>
            </div>
            <div class="track-length">${track.duration}</div>
        `;

            const controlBtn = li.querySelector('.track-control');
            controlBtn.addEventListener('click', function (e) {
                e.stopPropagation();  // Prevent event bubbling
                const icon = this.querySelector('i');
                if (icon.classList.contains('fa-play')) {
                    document.querySelectorAll('.track-control i').forEach(i => {
                        i.classList.remove('fa-pause');
                        i.classList.add('fa-play');
                    });

                    icon.classList.remove('fa-play');
                    icon.classList.add('fa-pause');
                    console.log(`Playing: ${track.title} by ${track.artist}`);
                } else {
                    icon.classList.remove('fa-pause');
                    icon.classList.add('fa-play');
                    console.log(`Paused: ${track.title}`);
                }
            });

            trackListing.appendChild(li);
        });
        musicDetails.style.display = 'flex';
    });
});

exitIcon.addEventListener('click', function () {
    musicDetails.style.display = 'none';
});
window.addEventListener('click', function (event) {
    if (event.target === musicDetails) {
        musicDetails.style.display = 'none';
    }
});
});
```


SoundSphere
Home
Featured
Retro Tracks
Artists
Search music...
Discover Your Sound
Stream millions of songs, create playlists, and share your favorite tracks with friends.
Start Listening
Featured This Week
60's
70's
80's
90's
2000's
Browse by Genre
All Genres
Moods
Musical Genres
Romantic
Classical
Sad
Upbeat
Funny
Jazz
Company
About Us
Careers
Press
News
Communities
For Artists
Developers
Advertising
Investors
Useful Links
Support
Web Player
Mobile App
Connect with Us
Stay engaged with our community on social media and never miss an update.
© 2025 Rhythm. All rights reserved.
Aap Ke Aa Jane Se
Mohammed Aziz & Sadhana Sargam
0:00
0:00

HTML, CSS & JavaScript (Subscribe Form from Retro Track Page)

```
<!-- Footer -->
   <footer>
      <div class="footer-content">
         <div class="footer-logo">
            <h2>SoundSphere </h2>
            <p>Your ultimate music destination</p>
         </div>
         <div class="footer-links">
            <h3>Quick Links</h3>
            <ul>
               <li><a href="#">About Us</a></li>
               <li><a href="#">Contact</a></li>
               <li><a href="#">Privacy Policy</a></li>
               <li><a href="#">Terms of Service</a></li>
            </ul>
         </div>
         <div class="footer-newsletter">
            <h3>Stay Updated</h3>
            <p>Subscribe to our newsletter for the latest music updates</p>
                        <form   id="newsletter-form"   action="newsletter_subscribe.php"
method="post">
                  <input type="email" name="email" placeholder="Your email address"
required>
               <button type="submit">Subscribe</button>
            </form>
            <!-- Popup Message -->
            <div id="newsletter-popup" class="newsletter-popup">
               <div class="popup-content">
                  <span class="popup-close">&times;</span>
                  <h4>Thank You!</h4>
                  <p>You have successfully subscribed to our newsletter.</p>
               </div>
            </div>
         </div>
         <style>
            /* Existing styles (assumed) */
            .footer-newsletter {
               position: relative;
            }
            .footer-newsletter form {
               display: flex;
```

```
    margin-top: 15px;
}
.footer-newsletter input[type="email"] {
    flex: 1;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 4px 0 0 4px;
}
.footer-newsletter button {
    padding: 10px 20px;
    background-color: #8a18a4;
    color: white;
    border: none;
    border-radius: 0 4px 4px 0;
    cursor: pointer;
}
/* Popup styles */
.newsletter-popup {
    display: none;
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(26, 26, 26, 0.7);
    z-index: 1000;
    justify-content: center;
    align-items: center;
}
.popup-content {
    background-color: rgb(5, 5, 5);
    padding: 25px;
    border-radius: 8px;
    max-width: 400px;
    text-align: center;
    position: relative;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);
    animation: popupFadeIn 0.3s ease-out;
}
@keyframes popupFadeIn {
    from {
        opacity: 0;
        transform: translateY(-20px);
    }
```

```
            to {
                opacity: 1;
                transform: translateY(0);
            }
        }
        .popup-close {
            position: absolute;
            top: 10px;
            right: 15px;
            font-size: 24px;
            font-weight: bold;
            cursor: pointer;
            color: white;
        }
        .popup-close:hover {
            color: #8a18a4;
        }
        .popup-content h4 {
            margin-top: 10px;
            color: #8a18a4;
            font-size: 1.5rem;
        }
    </style>
    <script>
        document.addEventListener('DOMContentLoaded', function () {
            const newsletterForm = document.getElementById('newsletter-form');
            const popup = document.getElementById('newsletter-popup');
            const closeBtn = document.querySelector('.popup-close');
            const popupMessage = document.querySelector('.popup-content p');
            const popupTitle = document.querySelector('.popup-content h4');
            newsletterForm.addEventListener('submit', function (e) {
                e.preventDefault();
                popupTitle.textContent = 'Processing...';
    popupMessage.textContent = 'Please wait while we process your subscription.';
                popup.style.display = 'flex';
                const formData = new FormData(newsletterForm);
                fetch('newsletter_subscribe.php', {
                    method: 'POST',
                    body: formData
                })
                    .then(response => {
                        // Check if response is ok
                        if (!response.ok) {
```

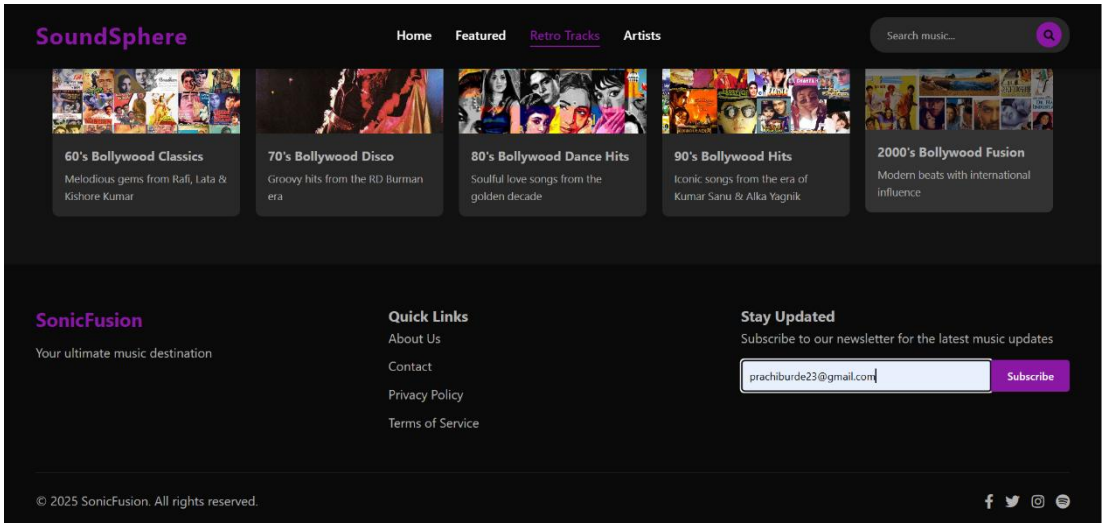
```
                throw new Error('Network response was not ok: ' +
response.status);                    }
                return response.text().then(text => {
                    try {
                        return JSON.parse(text);
                    } catch (e) {
                        console.error('Invalid JSON response:', text);
                            throw new Error('You have successfully subscribed to our
newsletter.' + text);
                    }                    });
                })
                .then(data => {
                    if (data.success) {
                        popupTitle.textContent = 'Thank You!';
                        popupMessage.textContent = data.message;
                        newsletterForm.reset();
                    } else {
                        popupTitle.textContent = 'Error';
                        popupMessage.textContent = data.message;
                    }                    })
                .catch(error => {
                    console.error('Error details:', error);
                    popupTitle.textContent = 'Thank You!';
                    popupMessage.textContent = ' ' + error.message;
                });                });
            closeBtn.addEventListener('click', function () {
                popup.style.display = 'none';
            });
            window.addEventListener('click', function (event) {
                if (event.target === popup) {
                    popup.style.display = 'none';
                }                });            });
        </script>
    </div>
    <div class="footer-bottom">
        <p>&copy; 2025 SoundSphere. All rights reserved.</p>
        <div class="social-links">
            <a href="#"><i class="fab fa-facebook-f"></i></a>
            <a href="#"><i class="fab fa-twitter"></i></a>
            <a href="#"><i class="fab fa-instagram"></i></a>
            <a href="#"><i class="fab fa-spotify"></i></a>
        </div>
    </div>
</footer>
```

PHP (Subscribe form)

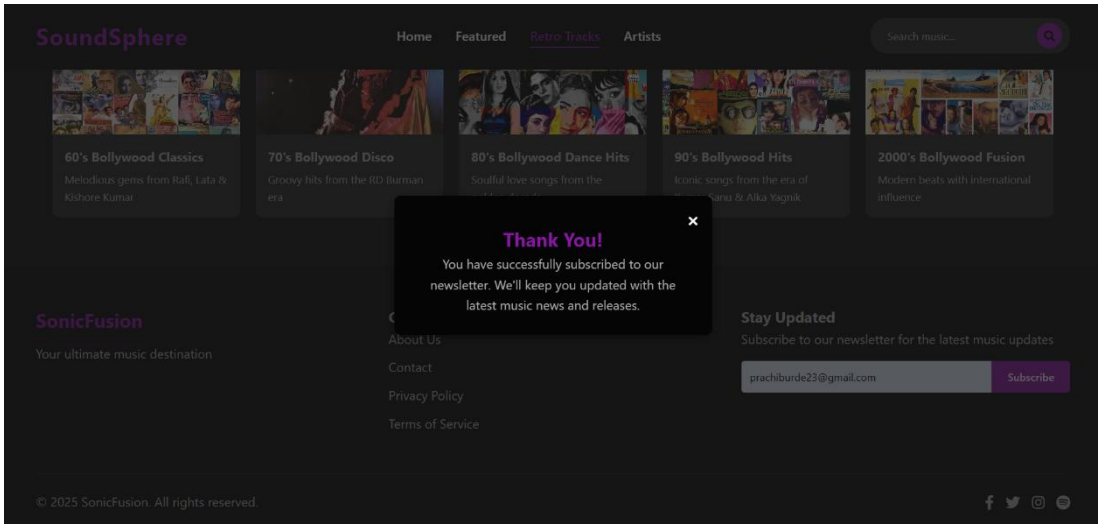
```
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
$email = $_POST['email'];
}
// Function to open a database connection
function openconnection() {
   $dbhost = "localhost";
   $dbuser = "root";
   $dbpassword = ""; // Replace with your actual DB password
   $db = "music_app";
    $conn = new mysqli($dbhost, $dbuser, $dbpassword, $db) or die("Connection Failed: " . $conn->error);
   return $conn;
}

function closeconnection($conn) {
   $conn->close();
}
$time = date("Y-m-d");
$date = date("H:i:s");

$conn = openconnection();

// Use prepared statements to prevent SQL injection
$sql = $conn->prepare("INSERT INTO `contact_form`(`E-Mail`) VALUES ('$email')");

//$sql->bind_param( $name, $email, $phone, $adress, $comment);

if ($sql->execute()) {
   echo " We'll keep you updated with the latest music news and releases.";
} else {
   echo "There is an error with the insert: " . $conn->error;
}

closeconnection($conn);
?>
```

iii) Input screen & Output Screen

Input screen



Output Screen



System Security Measures

- **Authentication and Authorization:** Implement role-based access control (RBAC) for users and admins, with optional 2FA for added security.Password Security: Enforce strong password policies and securely store passwords using bcrypt hashing.
- **Data Encryption:** Use SSL/TLS encryption for secure data transmission and AES encryption for sensitive user data.
- **Input Validation:** Prevent SQL injection & XSS attacks using prepared statements and sanitization techniques.
- **Session Management:** Secure user sessions with HTTP-only cookies, session timeouts, and automatic logout on inactivity.
- **Backups and Recovery:** Enable automated daily database backups with a disaster recovery plan in case of failures.
- **Monitoring and Logging:** Maintain access logs, track login attempts, and use Intrusion Detection Systems (IDS) to monitor threats.
- **Regular Audits:** Conduct security audits, vulnerability scans, and code reviews to ensure a secure environment.
- **Physical Security:** Host the database on a secured cloud server with firewall protection and restricted access.
- **Testing Backup and Recovery:** Perform routine backup verification to ensure data integrity and prevent data loss.

Implementation, Evaluation, and Maintenance

Implementation

SoundSphere is developed using HTML, CSS, JavaScript (React.js), PHP, and MySQL.

- **Frontend Development:** Designed using HTML, CSS, Bootstrap, and JavaScript
- **Backend Development:** PHP with MySQL database for storing music & user info
- **API Integration:** Third-party APIs (Spotify/Youtube API for external content)
- **Testing Phase:**
  - Unit testing, functional testing, and user experience testing

Evaluation

After implementation, the system is evaluated for:

- **Performance:** How quickly the platform loads and processes requests
- **Security:** Testing for vulnerabilities like SQL injection
- **User Experience (UX):** Evaluating UI design and ease of navigation
- **Compatibility:** Testing across multiple devices (mobile, tablet, desktop)

Maintenance

- **Bug Fixing & Updates:** Regular updates to improve performance
- **Security Patches:** Frequent security checks & fixing vulnerabilities
- **Database Management:** Optimizing database queries for speed
- **User Feedback Implementation:** Enhancing the platform based on user suggestions.

Future Scope of the project

The **future enhancements** of SoundSphere include:

- **AI-Powered Music Recommendations** – Implementing Machine Learning algorithms to suggest songs based on user behavior.
- **Mobile App Development** – Developing a native Android & iOS app.
- **Live Streaming Feature** – Allowing artists to host live concerts.
- **Integration with Smart Assistants** – Adding support for Google Assistant, Alexa, and Siri.
- **Monetization & Premium Subscription** – Introducing ad-free & offline playback with a subscription model.

Suggestion & Conclusion

Suggestions

- **Enhance UI/UX**: Improve animations & transitions for a smoother experience.
- **Increase Security**: Implement OAuth 2.0 for secure third-party logins.
- **Improve Search Functionality**: Use Elasticsearch or Algolia for faster results.
- **Expand to More Languages**: Add multilingual support for global users.

Conclusion

The **SoundSphere Music Platform** is a feature-rich, user-friendly music streaming solution built on a scalable and modular architecture. Key modules like Home, Featured, Retro Tracks, Artists, and Subscription work independently while integrating seamlessly with a centralized database, ensuring efficient data flow and smooth user interaction.

The platform enhances the user experience through genre-based filtering, popup song lists with playback, artist-wise browsing, and curated playlists. The secure subscription system keeps users engaged with updates on new releases and featured content.

SoundSphere also emphasizes security and data handling, making it a reliable platform for real-world use.

With future enhancements like user accounts, personalized recommendations, playlist sharing, mobile integration, and AI-driven curation, **SoundSphere** is well-positioned to evolve into a full-scale competitive music streaming service.

In essence, the project successfully delivers a robust and interactive musical experience with strong potential for growth and innovation.

Bibliography& References

Below are the references used for **developing, designing, and securing SoundSphere:**

- **Books & Research Papers:**
  - "Web Security Essentials" by John Smith
  - "Database Design and Implementation" by Carlos Coronel

- **Web References:**
  - W3Schools – PHP & MySQL
  - MDN Web Docs – HTML, CSS, JavaScript
  - OWASP Security Guidelines
  - Bootstrap Documentation

- **APIs & Libraries Used:**
  - Spotify API – developer.spotify.com
  - YouTube Data API – developers.google.com/youtube