**DOCUMENTATION OF MOVIE BOOKING SYSTEM PROJECT**

**Sound Entertainment: A Web-Based Music and Video Hosting Platform**

The **Sound Entertainment** project is a cutting-edge web-based application developed using **PHP** and **MySQL**. It is designed to provide a seamless platform for hosting, managing, and exploring a vast collection of music and videos across various categories.

**Key Features**

* **User-Friendly Interface**  
  Provides an intuitive and responsive interface, ensuring a smooth browsing experience for users on any device.
* **Dynamic Content Management**  
  Allows administrators to effortlessly manage media files, categories, and user accounts, ensuring up-to-date and well-organized content.
* **Rich Media Exploration**  
  Users can discover music and videos by **Album, Artist, Year, Genre, and Language**, making it easy to find their favorite content or explore something new.
* **Interactive User Experience**  
  Users can rate and review media content, providing feedback and contributing to a vibrant community.

**TABLE OF CONTENT**

* Problem Definition
* Customer Requirement Specification
* Project Plan
* Project Folder and Files Structure

**Problem Definition**

The **Sound Entertainment** platform is a web-based solution designed to host and organize music and video collections, addressing the challenges of managing and discovering diverse media content. This system offers an engaging, user-friendly experience for administrators and users alike, making it an efficient tool for both content management and entertainment.

**Key Objectives**

* **Content Categorization**  
  Organize music and videos by **Album, Artist, Year, Genre, and Language**, allowing users to easily browse and discover their favorite media.
* **Latest Updates**  
  Highlight the newest additions to the platform with dedicated sections for **latest music** and **latest videos**, marked with a "New" badge.
* **Ease of Access**  
  Provide a responsive design that enables users to access the platform from any device, ensuring a seamless experience.
* **Interactive Features**  
  Allow users to rate and review media, fostering a community-driven approach to content curation and feedback.
* **Admin Efficiency**  
  Equip administrators with tools to add, edit, and manage music/videos, categories, and user accounts effectively.

**Benefits**

1. **Convenience**  
   Users can explore, review, and enjoy media from the comfort of their homes without any hassle.
2. **Comprehensive Media Management**  
   The system provides detailed information about each piece of content, including categories, ratings, and reviews.
3. **User Roles**
   * **Standard Users**: Discover, play, and review music/videos.
   * **Administrators**: Manage content and oversee platform operations.

**Customer Requirement Specifications**

### ****Overview****

The **Sound Entertainment Platform** is designed to offer an intuitive, feature-rich solution for hosting and organizing a collection of music and videos. It caters to two primary user groups: **Registered Users** and **Administrators**, with distinct roles and functionalities.

### ****Features****

#### **1. User Registration and Login**

* Enables users to create accounts with secure login functionality.
* Validations ensure unique credentials and personal details for user accounts.

#### **2. Content Discovery**

* Users can explore music and videos by **Album, Artist, Year, Genre, and Language**.
* Latest music and videos are highlighted with a "New" badge for easy identification.

#### **3. Content Details**

* Provides comprehensive details, including **ratings, reviews**, and **trailers** (for videos).
* Enables users to decide on content based on curated information.

#### **4. Ratings and Reviews**

* Users can rate and review music/videos, contributing to a community-driven feedback system.

#### **5. Administrator Dashboard**

* A centralized platform for managing the entire library of music and videos.

#### **6. Content Management**

* **Add, Edit, or Delete** music and video files efficiently.
* Manage categories like Album, Artist, Year, Genre, and Language for seamless navigation.

#### **7. User Management**

* Admin can view, edit, or delete user accounts.

#### **8. Special Concessions (Optional)**

* Discounts for specific age groups or categories, enhancing user engagement.

#### **9. Responsive Design**

* Tailwind CSS ensures an adaptable interface for various devices and screen sizes.

**Project Plan**

This plan outlines the development of the **Sound Entertainment Platform** over one month, utilizing HTML, CSS, JavaScript, jQuery, AJAX, and Tailwind CSS.

**Week 1: Setup and Design**

* **Requirement Analysis:** Finalize platform features and define the database schema, including categories such as Albums, Artists, Genres, and Languages.
* **Environment Setup:** Install and configure necessary tools, including Tailwind CSS.
* **Design:** Create wireframes and static pages for Home, Login/Registration, Media Details, and Admin Dashboard.
* **Database:** Design and implement tables for users, media content, categories, reviews, and ratings.

**Week 2: User Interface Development**

* **User Pages:**
  + Build responsive layouts using Tailwind CSS.
  + Add interactive features with jQuery and AJAX for seamless user experience (e.g., search, filter, and sorting).
* **Admin Dashboard:**
  + Design an intuitive dashboard interface for content and user management.
* **Authentication:** Implement secure login and registration functionality for both users and admins.

**Week 3: Backend Integration**

* **User Functions:** Develop PHP scripts for user profile management, reviews, and ratings.
* **Media Management:** Enable content upload, categorization, and search functionality for users.
* **Admin Functions:**
  + Implement CRUD operations for media, categories, and user accounts.
  + Develop tools for managing platform content efficiently.
* **AJAX Integration:** Ensure smooth asynchronous updates for interactive features like live search and content updates.

**Week 4: Testing and Deployment**

* **Testing:** Conduct unit and integration testing to ensure functionality, responsiveness, and cross-browser compatibility.
* **Feedback and Optimization:** Address identified bugs and optimize system performance.
* **Deployment:** Launch the platform on a web server and provide detailed setup and usage documentation.

**Project Folder Structure**

**1. admin (Backend Management)**

This folder contains the logic for the administration panel of the music and video system, with controllers for managing different content (albums, artists, genres, music, users, etc.). It also includes views and routes that define how each section is handled in the admin panel.

* **controller**: PHP files responsible for the logic of adding, modifying, removing, and viewing data related to albums, artists, genres, music, videos, and users.
  + **dashboard.controller.php**: Logic for displaying the dashboard.
  + **view\_user.controller.php**: Logic to view user information.
* **includes**: Contains common elements that are included in the pages, such as the header.
  + **header.admin.php**: The header for the admin panel pages.
* **route**: Organized routes (folders for different categories like albums, artists, genres, music, users, and videos).
  + **album**, **artist**, **genre**, **music**, **user**, **video**: Each folder has operations like adding, modifying, removing, and viewing the respective entity. For example:
    - **add.album.php**: Adds a new album.
    - **modify.album.php**: Modifies an existing album.
    - **remove.album.php**: Removes an album.
    - **view.album.php**: Views album details.
  + **index.php**: The main entry point for the admin panel.

**2. assets (Static Resources)**

This folder contains all static resources, such as images and media files used throughout the system.

* **images**: Various images for albums, artists, music, and videos used on the website, like banners and album artwork.
* **media**: Contains subfolders for images, songs, and videos.
  + **images**: Images used for artists, album covers, and video thumbnails.
  + **songs**: Audio files (in MP3 or M4A formats).
  + **videos**: Video files (MP4 format).

**3. controller (User-side Interactions)**

These files handle the user-facing logic, including user authentication, booking, commenting, music and video fetch operations, and submitting comments or replies.

* **check\_login\_status.php**: Checks if a user is logged in.
* **delete\_comment.controller.php**: Handles deleting a comment.
* **fetch\_comments.controller.php**: Fetches comments for a particular item (music/video).
* **fetch\_music.controller.php**: Fetches music data.
* **fetch\_video.controller.php**: Fetches video data.
* **get\_video.controller.php**: Fetches specific video information.
* **login.controller.php**: Manages user login.
* **post\_comment.controller.php**: Allows posting comments on music or videos.
* **register.controller.php**: Handles user registration.
* **search.controller.php**: Handles searching functionality.
* **submit\_reply.controller.php**: Handles submitting replies to comments.

# E-R Diagram of Database

**4. frameworks**

This folder contains third-party libraries, such as JavaScript frameworks.

* **jquery**: Contains the jQuery library used for handling dynamic interactions on the website.
  + **jquery.min.js**: The minified version of the jQuery library.

**5. includes (Common Includes)**

Contains files for reusable components like the website header and footer.

* **header.php**: Contains the header used across user-facing pages.
* **footer.php**: Contains the footer used across user-facing pages.

**6. routes (Routing Files)**

Contains PHP files that define the routes for the various parts of the website, including user registration, login, music, videos, and categories.

* **categories.php**: Handles the category routes for music, videos, and other content.
* **login.php**: Route for the user login process.
* **logout.php**: Route for the user logout process.
* **music.php**: Handles music-related routes.
* **musics.php**: Contains music-specific routes (likely a collection of music).
* **register.php**: Handles user registration routes.
* **video.php**: Handles video-related routes.
* **videos.php**: Contains video-specific routes.

**7. Project Root Files**

* **.gitattributes**: A Git configuration file for managing Git attributes for the project.
* **basic-structure.txt**: A text file that describes the basic structure of the project.
* **config.php**: Configuration file for connecting to the database and setting up other settings for the application.
* **database.sql**: SQL file to set up the initial database schema (tables, relationships, etc.).
* **index.php**: The main entry point for the front-end of the website.
* **query.sql**: SQL queries for setting up or interacting with the database.