Front-End UI/UX Mini Project

Project Title: "**Music Playlist Organizer**"

• **Submitted By**:
Keerthi M (2462335) – keerthi.m@btech.christuniversity.in

Shalini V (2462363) – shalini.venkateshwaran@btech.christunivesity.in

Ayan Lamba (2462321) - ayan.lamba@btech.christuniversity.in

• Course: UI/UX Design Fundamentals

• Instructor Name: Ms.Nagaveena

• Institution: Christ University

• Date of Submission: 25/09/2025

Project Overview

Create and interactive platform that helps users organize, browse, and manage their music playlists efficiently. Users can categorize songs, filter by genre, mood, or artist, and view detailed playlist information.

Abstract

The **Music Playlist Organizer** is a web-based application designed to help users efficiently manage their music playlists. It allows users to categorize songs by genre, mood, artist, and duration, and provides features such as playlist listings with cover images, search and filter functionality, and a detailed playlist view for editing and playback. Built using HTML, CSS, JavaScript, Bootstrap, and jQuery, the platform is fully responsive for both desktop and mobile use. This tool offers a simple yet powerful solution for personalized music organization.

Objectives

- To develop a web-based platform that allows users to organize, browse, and manage music playlists effectively.
- To implement playlist categorization based on genre, artist, mood, and duration for easier navigation and filtering.
- To provide detailed playlist views, including song information and options to play, edit, or remove tracks.
- To ensure responsive design so the platform works seamlessly on both desktop and mobile devices.
- To use front-end technologies like HTML, CSS, JavaScript, Bootstrap, and jQuery for a user-friendly interface.

Scope of the Project

The platform will focus on frontend playlist management only—including listing, filtering, and basic CRUD (Create, Read, Update, Delete) operations for songs within playlists.

- It will support search and filter functionality based on multiple criteria (genre, mood, artist, and duration).
- The playlist interface will include visual elements like cover images, titles, and short descriptions.
- User authentication and music playback streaming (e.g., integrating with Spotify or YouTube) are not included in this phase.

Tools & Technologies Used

Tool/Technology	Purpose
• **HTML5**	• Markup and content structure
• **CSS3**	• Styling and layout management
• **VS Code**	• Code editor
• **Chrome DevTools**	• Testing and debugging
• **JavaScript**	• Interactivity and dynamic behaviour
• **Bootstrap**	• Responsive design
• **jQuery**	• Simplified javascript operations

HTML Structure Overview

- Header: Site name/logo and navigation
- Filters: Search bar and dropdowns for genre, mood, etc.
- Playlists Section: Cards showing playlists with images and details.
- Playlist View (Optional): Detailed view of songs in a playlist.
- Footer: Copyright and basic info.
- External Links: Bootstrap, jQuery, custom CSS/JS.

CSS Styling Strategy

- Custom CSS used for colors, spacing, and layout refinements.
- Class-based styling applied for consistency and reusability.
- Responsive design ensured through media queries.
- External stylesheet (styles.css) linked to keep structure and style separate.

JavaScript Strategy

- Handles **interactivity** like search, filter, and playlist updates.
- Event **listeners** for user actions (e.g., button clicks, dropdown changes).
- DOM **manipulation** used to update playlist content dynamically.
- Separate **(script.js) file** used for better code organization.

Bootstrap Usage

- Grid system used for responsive layout (e.g., playlist cards).
- Pre-built components like buttons, cards, navbar, and forms.
- Utility classes used for margin, padding, and text formatting.
- Ensures mobile-first design out of the box.

jQuery Usage

- Simplifies DOM manipulation and element selection.
- Used for handling events like filtering playlists.
- Makes AJAX calls easier if connecting to APIs later.
- Helps toggle content or apply animations smoothly.

Key Features

Feature	Description
Responsive Design	Adapts seamlessly to all screen sizes
Search & Filter	Filter playlists for easier navigation
Dynamic content handling	Update content without page reloading
Modular design	Structured html, css, js
External libraries integration	Leverages bootstrap for UI components

Challenges Faced and Solutions

Challenge	Solution
Managing dynamic filtering	Use JS/jQuery to handle filters
Ensuring responsive design	Utilize bootstrap's grid system
Handling large playlists without slowing down the interface	Implement pagination
Cross-browser compatibility issues	Test across tools like Chrome DevTools & BrowserStack

Outcome

- Developed a responsive web app for managing music playlists.
- Enabled filtering and searching by genre, mood, artist, and duration.
- Implemented dynamic playlist and song management with real-time updates.
- Created an intuitive and visually appealing user interface.
- Demonstrated effective use of HTML, CSS, JavaScript, Bootstrap, and jQuery.
- Provided a scalable foundation for future music-related features.

Future Enhancements

- Add user authentication for personalized playlist management.
- Integrate with music streaming APIs (e.g., Spotify, YouTube) for real-time playback.
- Implement playlist sharing features via social media or direct links.
- Introduce drag-and-drop functionality to reorder songs in playlists.
- Add smart recommendations based on user preferences and listening history.
- Develop a dark mode option for better user experience in low light.
- Enable offline access to playlists using browser storage or caching.

Sample Code

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Music Playlist Organizer</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

  <!-- jQuery -->
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

  <!-- Custom CSS -->
  <link rel="stylesheet" href="style.css">
</head>
<body>

<div class="container py-4">
  <h1 class="text-center mb-4">🎵 Music Playlist Organizer</h1>

  <!-- Search & Filter -->
  <div class="row mb-3">
    <div class="col-md-6">
      <input type="text" id="searchInput" class="form-control" placeholder="Search by title, artist, or genre...">
    </div>
```

```
    <div class="col-md-6">
      <select id="filterGenre" class="form-control">
        <option value="">Filter by Genre</option>
        <option value="Pop">Pop</option>
        <option value="Rock">Rock</option>
        <option value="Hip-Hop">Hip-Hop</option>
        <option value="Classical">Classical</option>
      </select>
    </div>
  </div>

  <!-- Playlist Listing -->
  <div class="row" id="playlistContainer"></div>

  <!-- Playlist Details -->
  <div id="playlistDetails" class="mt-4">
    <button class="btn btn-secondary mb-3" id="backBtn">⬅ Back to Playlists</button>
    <h3 id="playlistTitle"></h3>
    <p id="playlistDesc"></p>
    <div id="songsList"></div>
  </div>
</div>

<!-- Bootstrap JS -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<!-- Custom JS -->
<script src="script.js"></script>
</body>
</html>
```

CSS

```
body {
  background: #f4f6f9;
  font-family: Arial, sans-serif;
}

.playlist-card {
  transition: 0.3s;
  cursor: pointer;
}

.playlist-card:hover {
  transform: scale(1.03);
  box-shadow: 0 4px 15px rgba(0,0,0,0.15);
}

.playlist-cover {
  height: 180px;
  object-fit: cover;
}

#playlistDetails {
  display: none;
}

.song-item {
  border-bottom: 1px solid #ddd;
  padding: 12px 0;
  display: flex;
  flex-direction: column;
```

```
}

.song-meta {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

audio {
  margin-top: 5px;
  width: 100%;
}
```

JAVASCRIPT

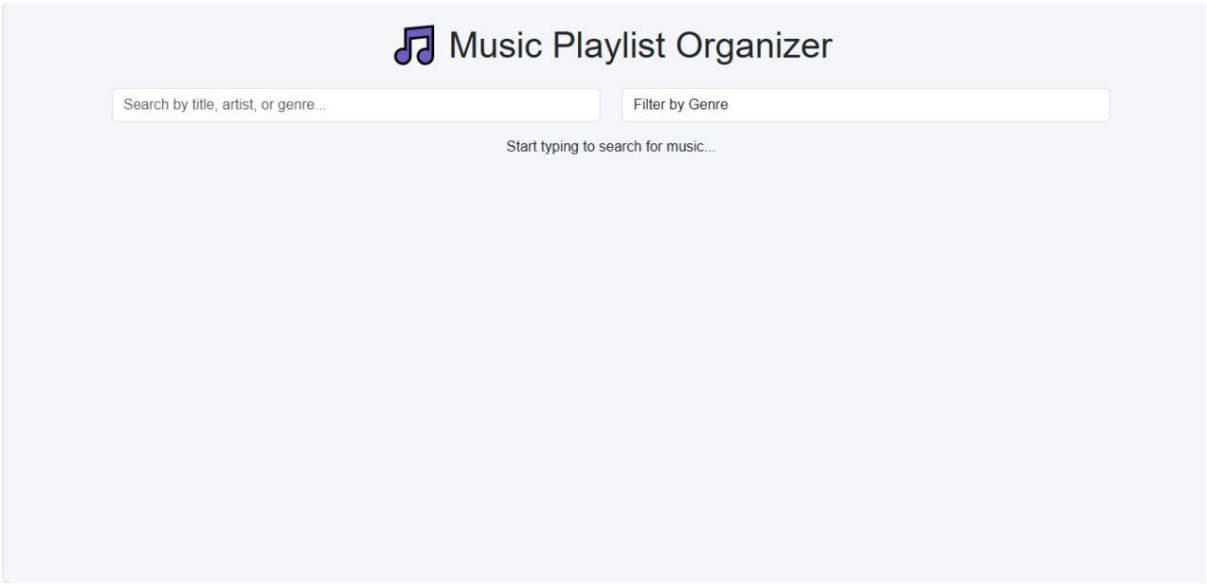
```
// Sample playlist data is no longer needed since we are using an API.
// Remove the 'const playlists = [...];' array.

// Function to display playlists (now search results from iTunes)
let currentAudio = null;

function displayPlaylists(filterText = "") {
  $("#playlistContainer").empty();

  if (filterText.length < 3) {
    $("#playlistContainer").html("<p class='text-center'>Start typing to search for music...</p>");
    return;
  }
```

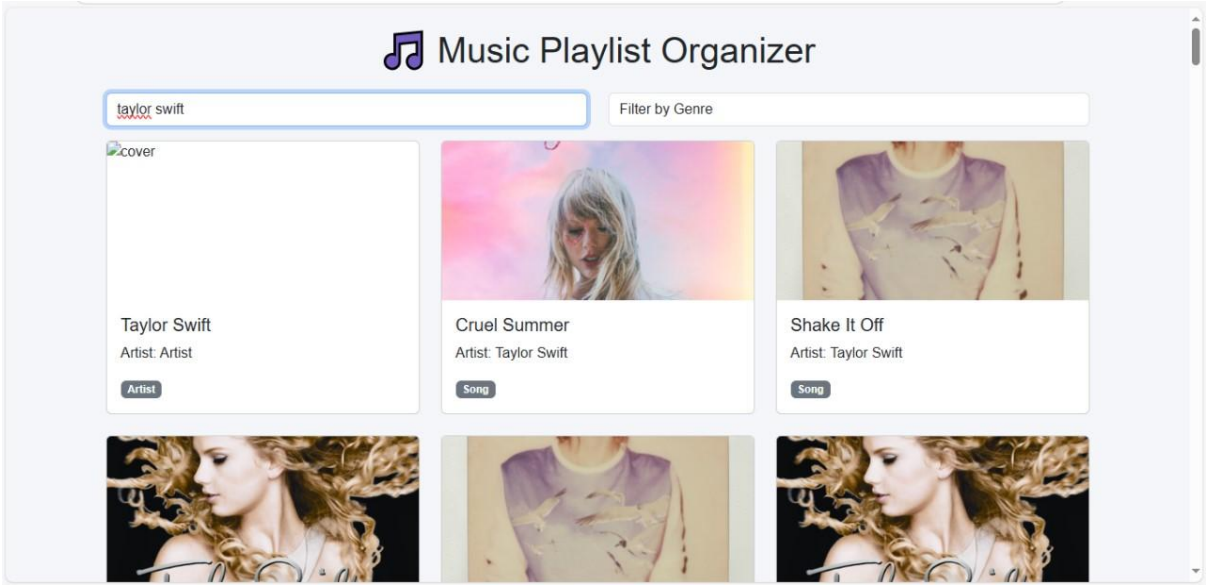
```
  const searchUrl =
https://itunes.apple.com/search?term=${encodeURIComponent(filterText)}&entity=song,all
Artist&limit=25;


  $.ajax({
    url: searchUrl,
    dataType: "jsonp", // Use JSONP for cross-domain requests
    success: function(data) {
      if (data.results && data.results.length > 0) {
        data.results.forEach(item => {
          const isSong = item.kind === "song";
          const title = isSong ? item.trackName : item.artistName;
          const artist = isSong ? item.artistName : "Artist";
          const cover = item.artworkUrl100 ? item.artworkUrl100.replace('100x100', '400x400') :
"https://via.placeholder.com/400";
          const audioPreview = item.previewUrl;
          const type = isSong ? "Song" : "Artist";


          const card = `
            <div class="col-md-4 mb-4">
              <div class="card playlist-card" data-preview-url="${audioPreview}">
                <img src="${cover}" class="card-img-top playlist-cover" alt="cover">
                <div class="card-body">
                  <h5 class="card-title">${title}</h5>
                  <p class="card-text">Artist: ${artist}</p>
                  <span class="badge bg-secondary">${type}</span>
                </div>
              </div>
            </div>
          `;
          $("#playlistContainer").append(card);
```

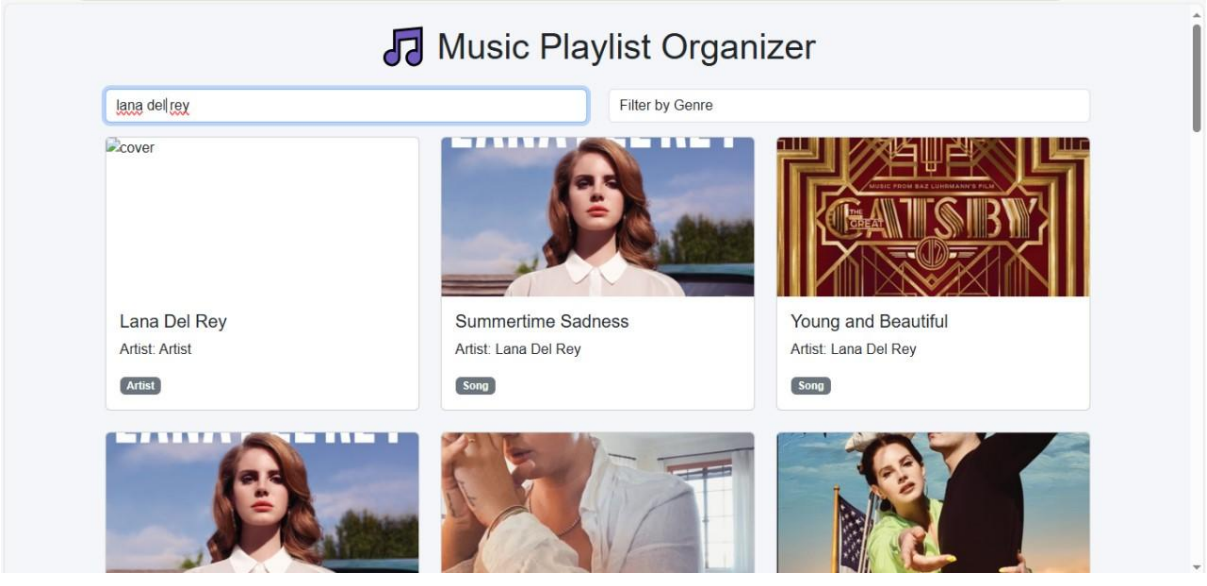
```
      });
    } else {
      $("#playlistContainer").html("<p class='text-center'>No results found.</p>");
    }
  },
  error: function(jqXHR, textStatus, errorThrown) {
    console.error('Error fetching data:', textStatus, errorThrown);
    $("#playlistContainer").html("<p class='text-center text-danger'>Error loading music. Please try again.</p>");
  }
 });
}

// Show details of a playlist - this function is no longer used.
// Remove the 'showPlaylistDetails(id)' function.

$(document).ready(function() {
 // Initial call to show a message
 displayPlaylists();

 // Search
 $("#searchInput").on("keyup", function() {
  const value = $(this).val().toLowerCase();
  displayPlaylists(value);
 });

 // Filter by Genre - this is no longer used.
 // Remove the '$("#filterGenre").on("change", function() { ... });' block.

 // Click playlist card to play audio
 $(document).on("click", ".playlist-card", function() {
```

```
    // Stop the previous audio if it's playing
    if (currentAudio) {
      currentAudio.pause();
      currentAudio.currentTime = 0;
    }

    const previewUrl = $(this).data("preview-url");
    if (previewUrl && previewUrl !== "null") {
      currentAudio = new Audio(previewUrl);
      currentAudio.play();
    } else {
      alert("No audio preview available for this item.");
    }
  });

  // Back button - this is no longer used.
  // Remove the '$("#backBtn").click(function() { ... });' block.
});
```

Screenshots of Final Output







Conclusion

The Music Playlist Organizer project successfully demonstrates how an interactive, user-friendly web application can simplify music management. By combining responsive design, efficient filtering, and dynamic content updates, the platform enhances the user experience for organizing and enjoying playlists. The use of modern web technologies like HTML, CSS, JavaScript, Bootstrap, and jQuery ensures both functionality and accessibility across devices. This project lays a strong foundation for further improvements and integration with music streaming services.

References

• L&T LMS : https://learn.lntedutech.com/Landing/MyCourse