L&T EduTech

Front-End UI/UX Mini Project
Project Submission Report

1. Title Page

Project Title: Music Playlist Organizer

Submitted By:

Umeshwar Kumar Pandit – 2462166 - umeshwar.kumar@btech.christuniversity.in

Sheryn Anand – 2462148 - sheryn.anand@btech.christuniversity.in

Aryan Sharma – 2462047 – aryan.sharma@btech.christuniversity.in

Course: UI/UX Design Fundamentals

Instructor Name: ***Dhiraj Alate***

Institution: Christ University

Date of Submission: 26/09/2025

2. Abstract

This project presents the design and development of an Interactive Music Playlist Organizer, a sophisticated front-end web application crafted to deliver a seamless and visually engaging user experience. The primary goal was to create a platform where users can effortlessly browse, filter, and interact with their music collections, inspired by the intuitive design of modern streaming services like Spotify. The application is built upon a robust foundation of **HTML5, CSS3, JavaScript (ES6), and the jQuery library**, with the **Bootstrap 5 framework** ensuring full responsiveness across all devices. The final outcome is a high-fidelity prototype that not only meets all functional requirements, including live audio playback, but also elevates the user experience through advanced features like a custom glowing cursor and fluid, meaningful animations.

3. Objectives

The project was guided by the following core objectives:

- **Design a Modern User Interface:** To craft a visually compelling, dark-themed UI that is both aesthetically pleasing and highly intuitive, following contemporary design principles.
- **Develop a Fully Responsive Layout:** To ensure the application provides an optimal viewing and interaction experience on a wide range of devices, from mobile phones to desktops, utilizing Bootstrap 5.
- **Implement Dynamic Content Rendering:** To use JavaScript and jQuery to dynamically

generate and manipulate HTML content, such as the playlist grid and tracklists, based on a client-side data source.

- **Integrate Interactive Features:** To build core functionalities including a multi-criteria search and filter system (by genre, mood, artist), and an integrated HTML5 audio player for MP3 playback.
- **Enhance User Experience (UX):** To move beyond basic functionality by incorporating sophisticated micro-interactions, such as a custom glowing cursor and elegant fade-out animations on user actions, to make the application feel more responsive and alive.

4. Scope of the Project

The project's scope was intentionally focused on delivering a polished front-end experience. The boundaries are defined as follows:

- **Front-End Focus:** The application is entirely client-side. There is no backend integration, server-side logic, or database connectivity.
- **Client-Side Data:** All playlist and song data is managed within a local JavaScript object (data.js), acting as a mock database for the prototype.
- **Core Technologies:** The implementation relies exclusively on HTML, CSS, and JavaScript, supplemented by the established libraries of jQuery and Bootstrap to enhance development efficiency and ensure cross-browser compatibility.
- **Functionality:** The project includes playlist browsing, filtering, a detailed tracklist view, and functional audio playback. Features like creating new playlists or user authentication are outside the current scope.

5. Tools & Technologies Used

Tool/Technology	**Purpose**
HTML5	Utilized for structuring the web pages with semantic elements, ensuring accessibility and a logical document flow.
CSS3	Employed for all aspects of styling, from layout management and the dark-theme color scheme to advanced animations and custom cursor effects.
JavaScript (ES6)	The core language used for all dynamic functionality, including event handling, DOM manipulation, and managing the application's state.

jQuery	Leveraged to simplify and accelerate DOM traversal and manipulation, making the JavaScript code more concise and efficient.
Bootstrap 5	The foundational framework for building the responsive grid system, ensuring the user interface adapts seamlessly across all viewports.
VS Code	The primary code editor used for development.
Chrome DevTools	An essential tool for real-time testing, debugging JavaScript logic, and fine-tuning CSS styles.

6. HTML Structure Overview

The application is architected across two primary HTML files to separate concerns:

- **index.html:** Serves as the main dashboard, containing the structure for the navigation bar, the filter section, and a container (<section id="playlist-grid">) where the playlist cards are dynamically injected by JavaScript.
- **playlist.html:** The detail page, which includes structural elements for the playlist header, the HTML5 <audio> player, and a <tbody> where the tracklist is dynamically rendered.

Both pages utilize semantic HTML5 tags such as <nav>, <main>, and <header> to create a well-structured and accessible foundation.

7. CSS Styling Strategy

A comprehensive styling strategy was implemented to achieve the desired aesthetic and user experience:

- **External Stylesheet:** All custom styles are encapsulated in css/style.css to maintain separation of concerns.
- **Spotify-Inspired Dark Theme:** A sophisticated dark theme was created using a palette of black, grey, and white, creating a high-contrast and visually comfortable environment.
- **Responsive Layout:** Bootstrap's powerful grid system was used for the primary layout, while Flexbox was employed for finer alignment control within components.
- **Advanced UI Enhancements:**
    - **Custom Glowing Cursor:** A custom cursor was implemented using CSS pseudo-elements (::before) and JavaScript to create an engaging glowing dot that follows the

mouse.

  - **Transitions and Animations:** Smooth transition properties were applied to interactive elements to provide visual feedback. A custom CSS class (.clicked) was created to trigger an elegant "enlarge and fade-out" animation on playlist cards.
- **Mobile-First Considerations:** While not strictly a mobile-first build, responsive design principles were a priority, ensuring usability on smaller screens.

8. Key Features

Feature	**Description**
Dynamic Playlist Grid	The homepage dynamically displays a grid of playlist cards, each rendered from the central data source.
Multi-Criteria Filtering	Users can instantly filter the displayed playlists by typing in the search bar (searches titles and artists) or by selecting options from the Genre and Mood dropdown menus.
Detailed Playlist View	Clicking a playlist card navigates the user to a dedicated page that displays the playlist's cover art, description, and a complete, interactive tracklist.
Integrated Audio Player	The playlist page features a fully functional HTML5 audio player. Users can click on any song in the tracklist to play it, with the currently playing track being visually highlighted. The player also supports auto-play for the next song.
Elegant Click Animation	When a playlist card is clicked, it executes a smooth "enlarge and fade-out" animation before transitioning to the next page, providing a polished and professional user experience.

Custom Glowing Cursor	The default browser cursor is replaced with a custom-styled white dot with a soft glow, which subtly enlarges when hovering over interactive elements, adding a layer of sophisticated visual feedback.

9. Challenges Faced & Solutions

Challenge	**Solution**
Custom cursor disappearing on page navigation.	The initial JavaScript for the cursor only ran on index.html. This was solved by abstracting the cursor logic into its own dedicated file (js/cursor.js) and then including this script in both HTML files.
Click animation not being visible before page change.	The default link behavior caused the page to navigate instantly. The solution was to use jQuery to preventDefault() on the click event, apply the animation class, and then use setTimeout to delay the navigation until the CSS transition was complete.
Managing the "Now Playing" UI state.	To highlight the current song and update the play/pause icons correctly, event listeners (onplay, onpause, onended) were attached to the HTML5 audio element. These listeners trigger functions that dynamically add or remove CSS classes from the tracklist rows and buttons.

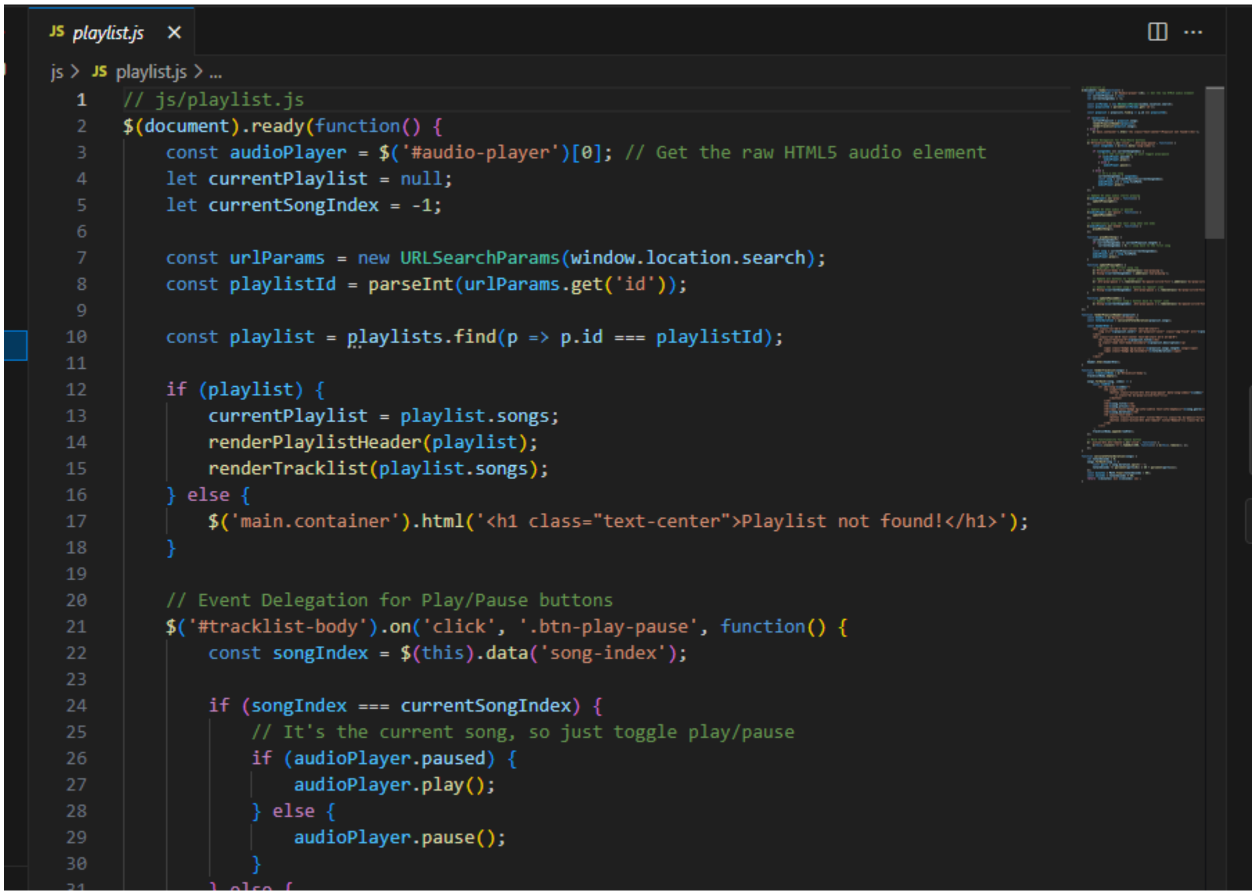
10. Outcome

The project successfully culminated in a fully functional and aesthetically polished front-end application. All initial objectives were met, resulting in a responsive, interactive, and visually engaging music organizer. The final product demonstrates a strong command of front-end technologies and a keen eye for user experience design. This project served as an excellent practical exercise in DOM manipulation, event handling, and creating a seamless multi-page experience using only client-side technologies.

11. Future Enhancements

- **Backend Integration:** Integrate a backend service (e.g., Node.js with Express) and a database (e.g., MongoDB) to allow users to create, save, edit, and delete their own persistent playlists.
- **User Authentication:** Implement a user login and registration system to personalize the experience.
- **Third-Party API Integration:** Fetch real playlist and song data from an external service like the Spotify API.
- **Advanced Audio Controls:** Enhance the audio player with features like a volume slider, a seek bar, and shuffle/repeat functionality.

12. Sample Code



```
// js/playlist.js
$(document).ready(function() {
    const audioPlayer = $('#audio-player')[0]; // Get the raw HTML5 audio element
    let currentPlaylist = null;
    let currentSongIndex = -1;

    const urlParams = new URLSearchParams(window.location.search);
    const playlistId = parseInt(urlParams.get('id'));

    const playlist = playlists.find(p => p.id === playlistId);

    if (playlist) {
        currentPlaylist = playlist.songs;
        renderPlaylistHeader(playlist);
        renderTracklist(playlist.songs);
    } else {
        $('main.container').html('<h1 class="text-center">Playlist not found!</h1>');
    }

    // Event Delegation for Play/Pause buttons
    $('#tracklist-body').on('click', '.btn-play-pause', function() {
        const songIndex = $(this).data('song-index');

        if (songIndex === currentSongIndex) {
            // It's the current song, so just toggle play/pause
            if (audioPlayer.paused) {
                audioPlayer.play();
            } else {
                audioPlayer.pause();
            }
```

Caption 1: JavaScript logic for the "enlarge and fade-out" click animation.
This snippet from js/script.js prevents the default link navigation, applies the animation class, and then redirects after a delay.

```
$('.playlist-link').on('click', function(e) {
  // Prevent the link from navigating immediately
  e.preventDefault();
```

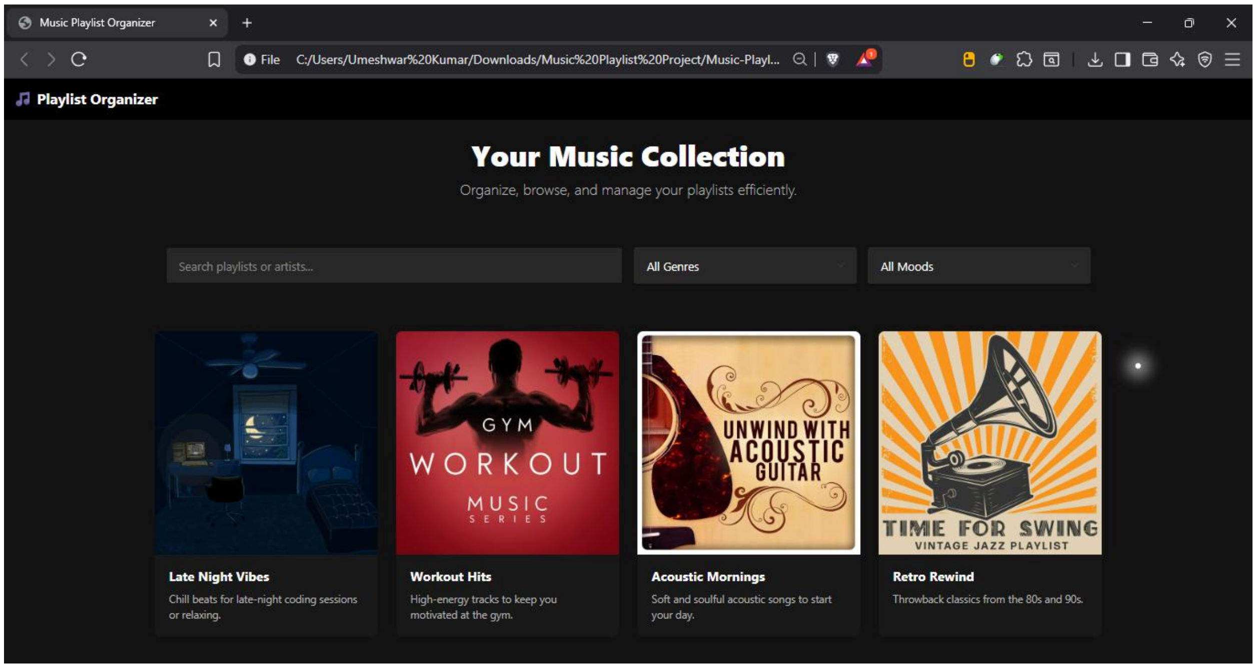
```
    const destination = $(this).attr('href');
    const card = $(this).closest('.playlist-card');

    // Add the animation class to the card
    card.addClass('clicked');

    // Wait for the animation to finish (400ms) before changing the page
    setTimeout(() => {
        window.location.href = destination;
    }, 400);
});
```

Caption 2: CSS for the custom glowing cursor and its interactive state.
This code from css/style.css hides the default cursor and styles the custom dot, its glow, and its enlargement on hover.

```
body {
    cursor: none;
}

.custom-cursor {
    position: fixed;
    left: 0;
    top: 0;
    width: 8px;
    height: 8px;
    background-color: #ffffff;
    border-radius: 50%;
    pointer-events: none;
    transform: translate(-50%, -50%);
    transition: width 0.2s, height 0.2s;
    z-index: 10000;
}

.custom-cursor::before {
    content: '';
    position: absolute;
    left: 50%;
    top: 50%;
    width: 30px;
    height: 30px;
    background-color: #ffffff;
    border-radius: 50%;
    transform: translate(-50%, -50%);
    filter: blur(10px);
    opacity: 0.6;
    z-index: -1;
}

.custom-cursor.active {
    width: 12px;
    height: 12px;
}
```

13. Screenshots of Final Output

Caption 1: Homepage / Main Playlist View on a Desktop Screen.



Music Playlist Organizer

File C:/Users/Umeshwar%20Kumar/Downloads/Music%20Playlist%20Project/Music-Playl...

Playlist Organizer

Your Music Collection

Organize, browse, and manage your playlists efficiently.

Search playlists or artists...

All Genres

All Moods

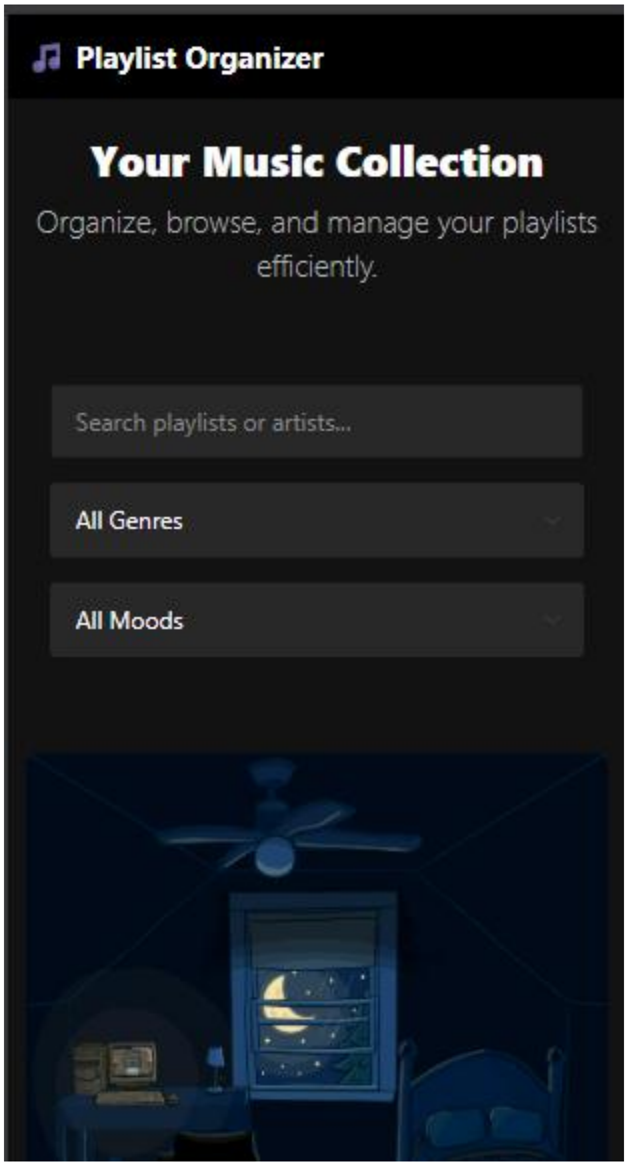
Late Night Vibes
Chill beats for late-night coding sessions or relaxing.

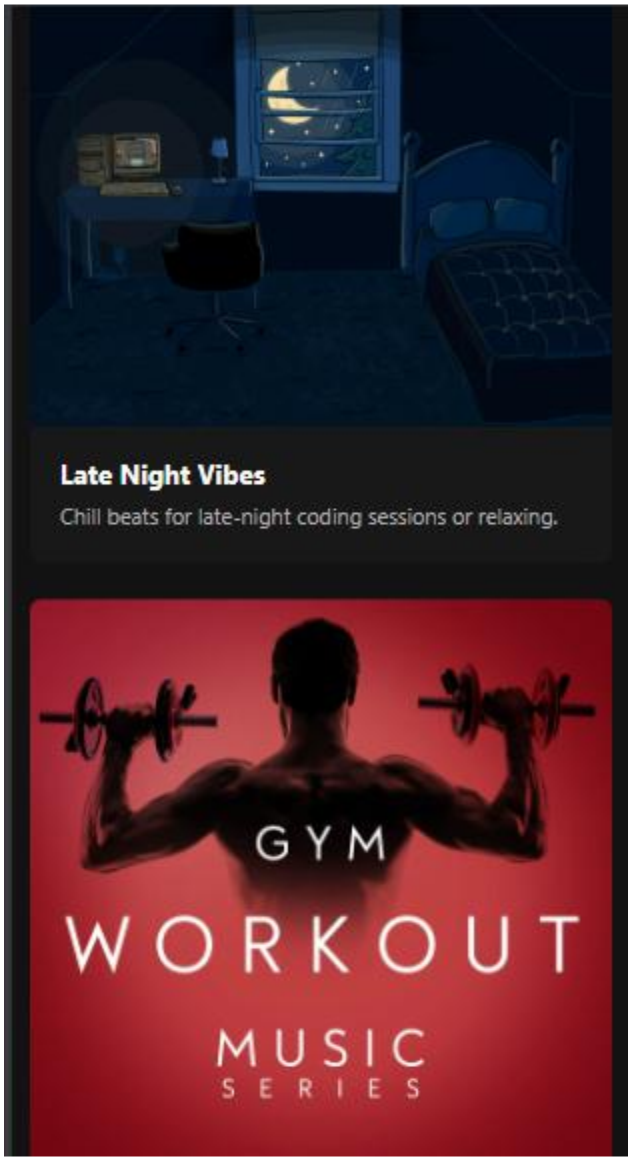
Workout Hits
High-energy tracks to keep you motivated at the gym.

Acoustic Mornings
Soft and soulful acoustic songs to start your day.

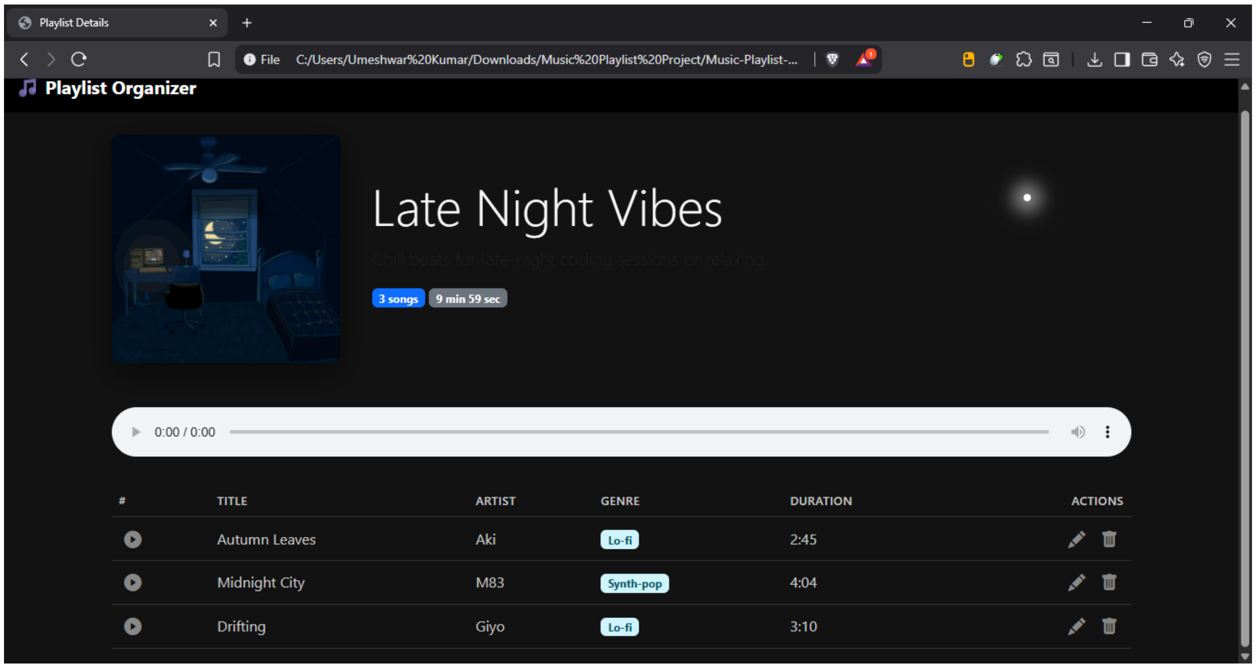
Retro Rewind
Throwback classics from the 80s and 90s.

Caption 2: Homepage View on a Mobile Device, demonstrating responsiveness.


Playlist Organizer
Your Music Collection
Organize, browse, and manage your playlists efficiently.
Search playlists or artists...
All Genres
All Moods



Caption 3: Playlist Detail Page, showing the tracklist and audio player.



Playlist Organizer

Late Night Vibes

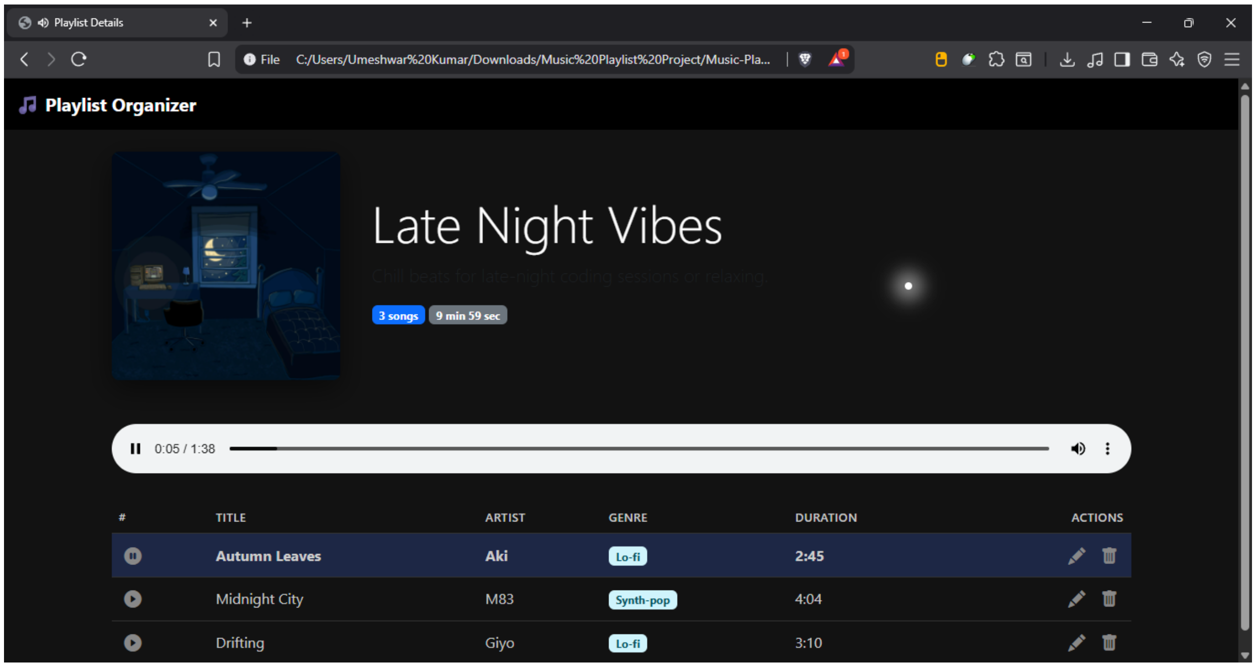
Chill beats for late-night coding sessions or relaxing

3 songs 9 min 59 sec

0:00 / 0:00

#	TITLE	ARTIST	GENRE	DURATION	ACTIONS
	Autumn Leaves	Aki	Lo-fi	2:45	
	Midnight City	M83	Synth-pop	4:04	
	Drifting	Giyo	Lo-fi	3:10	

Caption 4: A song playing, with the corresponding row highlighted.



Playlist Organizer

Late Night Vibes

Chill beats for late-night coding sessions or relaxing

3 songs 9 min 59 sec

0:05 / 1:38

#	TITLE	ARTIST	GENRE	DURATION	ACTIONS
	Autumn Leaves	**Aki**	Lo-fi	**2:45**	
	Midnight City	M83	Synth-pop	4:04	
	Drifting	Giyo	Lo-fi	3:10	

14. Conclusion

This mini-project was an immersive journey into modern front-end development, successfully blending technical implementation with user-centric design. The creation of the Interactive Music Playlist Organizer reinforced my skills in HTML, CSS, and JavaScript, while providing deep, practical insights into the power of libraries like jQuery and Bootstrap for rapid and robust development. Implementing features from a simple responsive layout to complex micro-interactions like the custom cursor has significantly enhanced my understanding of how thoughtful design and fluid functionality converge to create a truly compelling user experience.

15. References

- Bootstrap 5 Documentation: https://getbootstrap.com/docs/5.0/
- jQuery API Documentation: https://api.jquery.com/
- MDN Web Docs: https://developer.mozilla.org/