A FIELD PROJECT REPORT ON

SOUNDSCAPES

A Neon-Driven Music Experience

Submitted

In partial fulfillment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

by

Y. MADHU VEER	231FA04A36
N. MANISHA	231FA04A81
G. JYOTHI SWAROOP	231FA04B14
V. JOSHI	231FA04B64

Under the Guidance of

MR. K. PAVAN KUMAR

Assistant Professor, CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCHOOL OF COMPUTING AND INFORMATICS

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH

(Deemed to be University)

Vadlamudi, Guntur, Andhra Pradesh-522213, India

APRIL – 2025

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

CERTIFICATE

This is to certify that the field project entitled "SOUNDSCAPES" being submitted by Y. Madhu (231FA04A36), N. Manisha (231FA04A81), G. Jyothi Swaroop (231FA04B14), and V. Joshi (231FA04B64) in partial fulfilment of the Bachelor of Technology degree in the Department of Computer Science and Engineering, Vignan's Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur District, Andhra Pradesh, India, is a bonafide work carried out by them under my guidance and supervision.

The work embodied in this project is original and has not been submitted for the award of any other degree, diploma, or certificate earlier. The project work is carried out in accordance with the academic regulations and guidelines set by the university.

Guide

Project Review Committee

HoD, CSE

HoD
Dept. of Computer Science & Engine...
VFSTR Deemed to be Univers
VADLAMUDI - 522 213
Guntur Dist., A.P., India.

DECLARATION

Date: 26-04-2025

We hereby declare that the work presented in the field project titled "**SOUND SCAPES**" is the result of our own efforts and investigations.

This project is being submitted under the supervision of **Mr. K. Pavan Kumar , Assistant Professor, CSE** in partial fulfillment of the requirements for the Bachelor of Technology (B.Tech.) degree in Computer Science and Engineering at the Department of Computer Science and Engineering, Vignan's Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, India.

Y. MADHU VEER	231FA04A36	Signature: Y. MadhuVeer.
N. MANISHA	231FA04A81	Signature: N.manisha
G. JYOTHI SWAROOP	231FA04B14	Signature: G.JyothiSwaroop
V. JOSHI	231FA04B64	Signature: V. Joshi

Contents

1. INTRODUCTION:

Sound Scapes is a creative and interactive web-based music player project that transforms a simple listening experience into an engaging visual journey. Built using index.html and player.html, this application synchronizes music playback with dynamic color transitions, offering users a vibrant fusion of sound and light.

In today's digital era, most music players focus only on audio, missing the opportunity to stimulate multiple senses at once. Sound Scapes aims to bridge this gap by providing a rich, sensory experience where sound visually meets color and rhythm.

PROBLEM STATEMENT:

Despite the availability of many music players, most platforms focus solely on sound without leveraging the potential of real-time visual engagement. Users often desire a more immersive experience where the music they hear is complemented by dynamic visuals. **Sound Scapes** addresses this problem by introducing a lightweight, browser-based music player that visually reacts to the beat and mood of the audio being played, creating a multisensory environment.

CURRENT SYSTEM OVERVIEW:

Existing music players primarily:

- Offer basic audio controls like play, pause, and loop.
- Lack integration of dynamic, beat-synchronized visual effects.
- Provide static or minimal user interfaces without emotional engagement.
- Require heavy frameworks or complex setups for advanced visualizations.

Sound Scapes differentiates itself by offering an elegant & lightweight solution, using pure HTML5, CSS3, & JavaScript to create an engaging, real-time visual & audio experience without external dependencies.

PROPOSED SOLUTION:

The proposed solution, Sound Scapes, is a web application that enhances traditional music playback by incorporating:

- Dynamic Background Colors: Changes synchronized to song beats and mood palettes.
- Shuffle Play Feature: Randomly plays songs, ensuring a new experience each time.

- Animated Visual Elements: Includes a spinning disc, interactive loop button, and animated title.
- Theme Toggle: Allows switching between dark and light modes seamlessly.
- Interactive Playlist Management: Displays recently played songs and dynamically loads the playlist.

By combining these features, Sound Scapes transforms simple music listening into a visually enriched experience directly from the browser.

RELATED WORK:

Sound Scapes is built by integrating concepts and technologies from modern web development practices:

- HTML5 Audio API: Handles seamless audio playback in web browsers.
- CSS3 Animations and Transitions: Create smooth, real-time transformations and dynamic UI effects.
- JavaScript DOM Manipulation: Controls the playlist, player functions, beat-timed color transitions, and event-based actions (like theme switching and looping).
- UX/UI Research Insights: Support the idea that dynamic, responsive visual feedback significantly boosts user engagement and enjoyment while interacting with multimedia systems.

The system is inspired by advanced audio-visualizers but focuses on simplicity, accessibility, and efficiency.

2. SYSTEM REQUIREMENTS:

Sound Scapes is a web-based multimedia application optimized for lightweight performance and broad device compatibility. It is designed to run efficiently on systems with basic hardware configurations while delivering a rich audio-visual experience.

The design of Sound Scapes ensures that users do not require high-end systems to enjoy its features. By relying entirely on web technologies such as HTML, CSS, and JavaScript, the application is highly portable across different operating systems and devices. No external plugins or software installations are necessary, making it easy to access and operate directly through any modern web browser.

HARDWARE AND SOFTWARE REQUIREMENTS:

Hardware Requirements:

- Processor: Minimum 1 GHz (any modern processor)
- RAM: Minimum 2 GB (4 GB recommended for smooth multitasking)
- Storage: Minimal (only space for storing audio files and browser cache)
- Display: Any modern display supporting HTML5/CSS3 rendering
- Input Devices: Mouse, Keyboard (for interaction with the player)

Software Requirements:

- Operating Systems: Windows, macOS, Linux, or any OS supporting modern browsers.
- Web Browsers: Latest Versions of Chrome, Fire Fox, Edge, or Safari.(must support HTML5 and Java Script)
- Internet Access: Optional(only if audio files are streamed; local files can be played offline)

Functional Requirements:

- Play, pause, and loop audio files.
- Shuffle the playlist randomly.
- Dynamic background color transitions synchronized with beats.
- Recently played songs tracking.
- Light and dark mode theme toggle.
- Visual feedback for looping and active song.
- Responsive design to adapt to different screen sizes.

Non-Functional Requirements:

- Fast loading and minimal latency during interactions.
- Compatibility with all major modern web browsers.
- Smooth animations and transitions without heavy resource consumption.
- User-friendly interface with intuitive controls.

Technologies used:

- HTML5: Structuring the web content.
- CSS3: Styling, animations, and transitions.
- JavaScript: Functionality for audio playback, color transitions, and event handling.

3. SYSTEM ARCHITECTURE:

The Sound Scapes application follows a simple yet effective client-side architecture. It is entirely browser-based and built using HTML, CSS, and JavaScript, allowing it to operate independently without a back-end server.

The architecture is divided into two main components:

- User Interface Layer:

  Designed with HTML and styled using CSS, this layer provides the user with interactive elements such as the playlist, shuffle button, theme toggle, and the player window. Animations and dynamic effects enhance the user experience.
- Functional Logic Layer:

  Implemented using JavaScript, this layer controls the core functionality, such as audio playback, dynamic background color changes based on beats, looping features, and event handling for user actions.

All interactions occur locally within the browser, providing a fast and seamless experience without the need for server communication.

SYSTEM MODULES:

The Sound Scapes application is divided into the following functional modules:

Playlist Management Module

- Displays available songs.
- Manages recently played songs.
- Provides a shuffle option to play random songs.

Audio Player Module

- Plays, pauses, and loops audio tracks.
- Shows the current song title dynamically.
- Allows users to control playback through UI buttons.

Dynamic Background Module

- Changes background color based on the rhythm (beats) of the song.
- Applies different color palettes specific to each track.

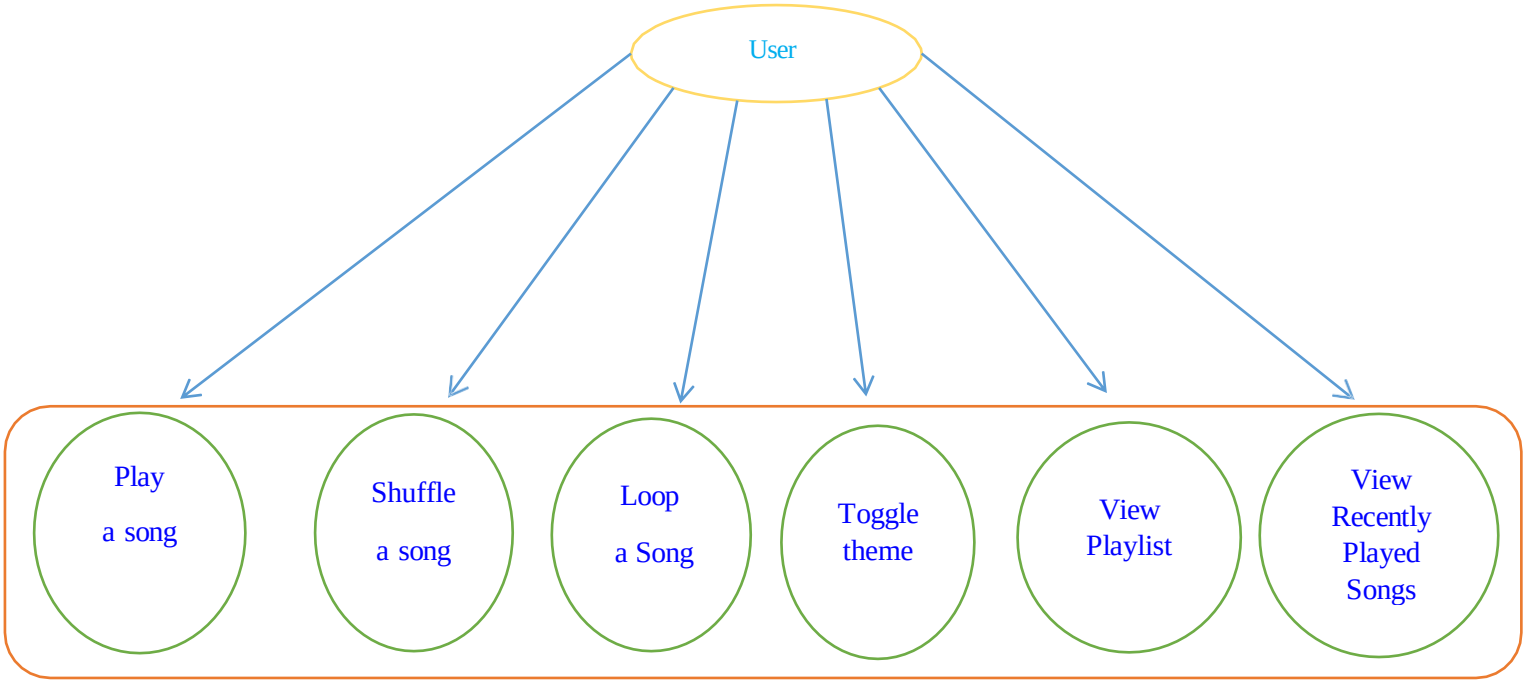
User Interaction Module

- Handles user events such as clicks on songs, shuffle button, loop toggle, and theme switching.
- Provides keyboard shortcuts (Space for play/pause, L for loop, T for theme toggle).

Theme Management Module

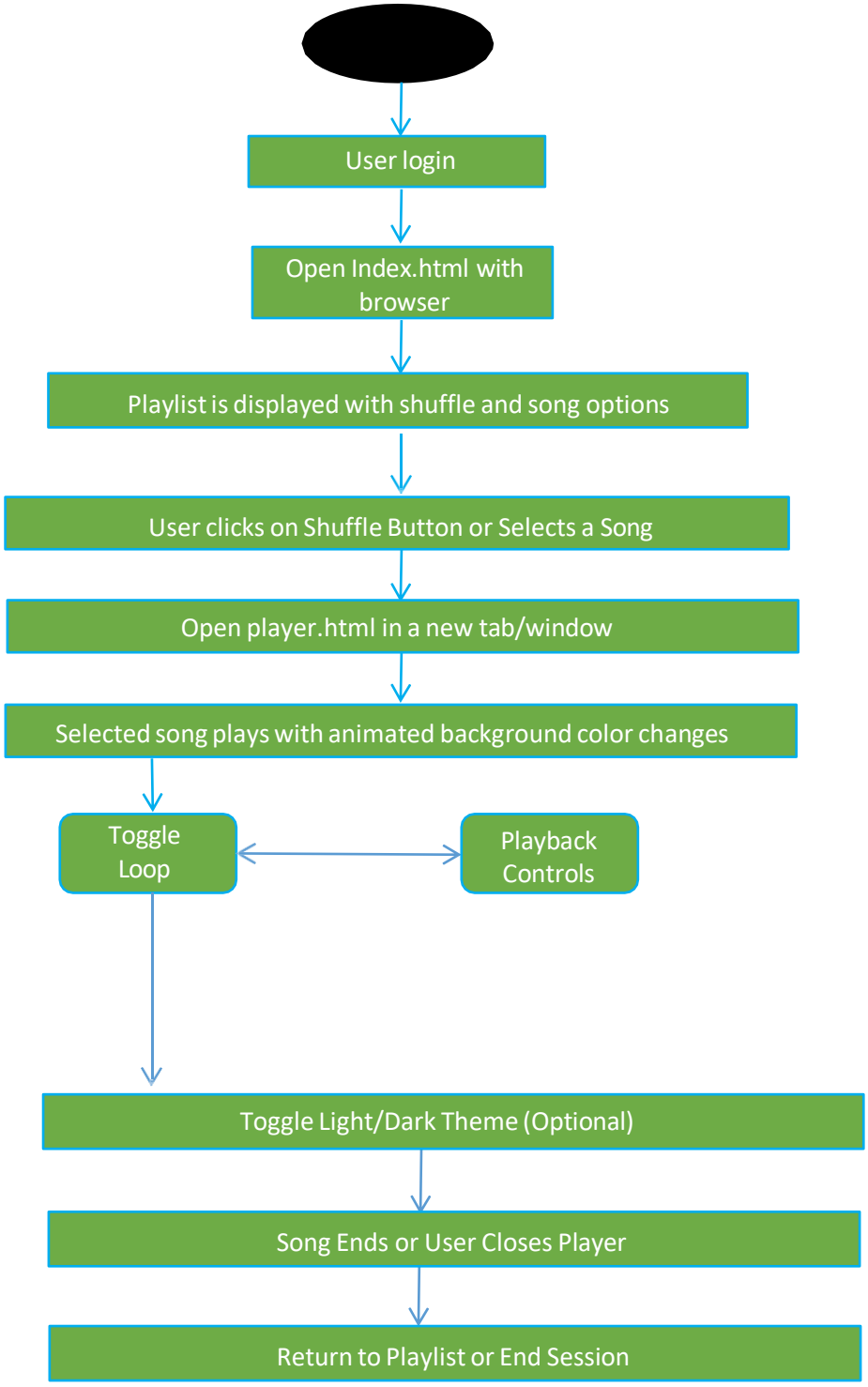
- Switches between dark mode and light mode.
- Automatically adjusts text and button colors based on background brightness.

UML Diagrams and Design:



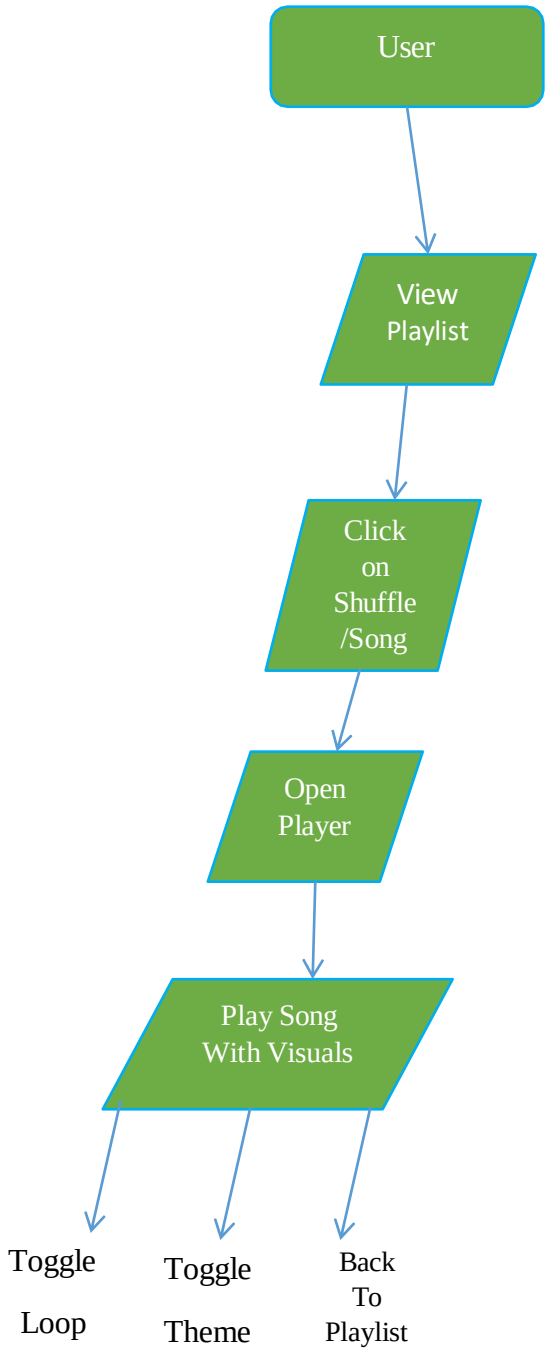
∴ User can access to play a song, Shuffle Songs, loop a song, Toggle theme, view playlist & Recently played songs on Soundscapes web pages

Use Case Diagram



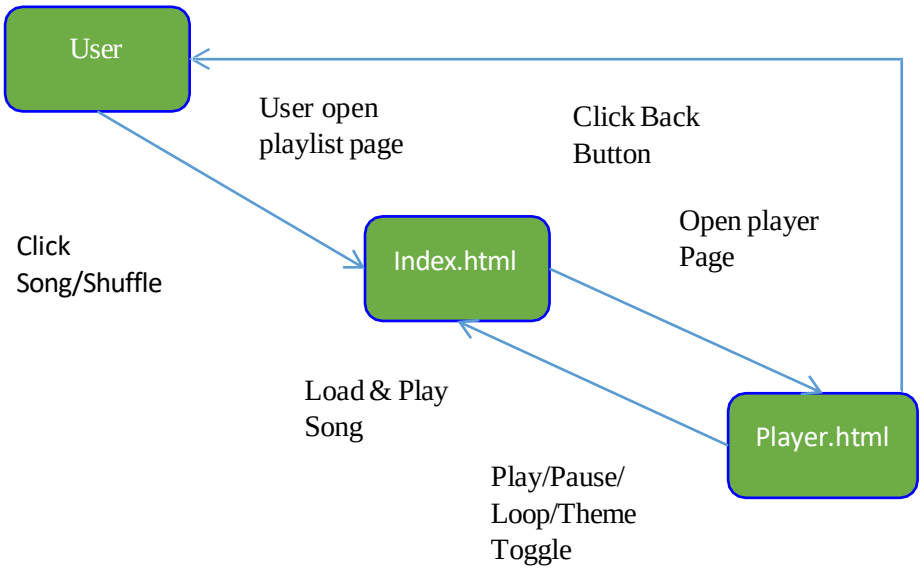
∴ use case diagram for Sound Scapes

ACTIVE DIAGRAM



∴ activity diagram for Sound Scapes

SEQUENCE DIAGRAM



∴ sequence diagram for
Sound Scapes

4. Implementation:

The Sound Scapes application was implemented using standard web technologies: HTML5, CSS3, and JavaScript.
The application follows a modular approach where each file (index.html, player.html) handles specific functionality.

index.html:
Acts as the main interface for the playlist and song selection. Users can shuffle songs or pick a specific track to play.

player.html:
Loads the selected song, plays audio, manages background color transitions synchronized with the beat, and provides interactive controls such as loop and theme toggling.

Dynamic behaviors such as playlist management, event handling, audio control, and visual effects are all managed through client-side JavaScript, ensuring a seamless and lightweight user experience.

Sample Code:

(index.html)

```
// index.html JavaScript

// List of available songs

const songs = [

  {

    name: "What The Life",

    src: "./audio/What The Life - SenSongsMp3.Co.mp3",

    colors: ['#ff0000', '#ffcc00', '#ffff00', '#00ff00', '#00ffff', '#ff00ff', '#ffffff'],

    beats: [200, 180, 220, 200, 190]

  },
```

```
  {
    name: "Aaya Re Toofan",
    src: "./audio/128-Aaya Re Toofan - Chhaava 128 Kbps.mp3",
    colors: ['#FF1493', '#FF4500', '#FFD700', '#00FF00', '#00BFFF', '#8A2BE2', '#FFFFFF'],
    beats: [150, 160, 140, 155, 145]
  }
  {
    name: "Kesariya",
    src: "./audio/kesariya.mp3",
    colors: ['#FF0000', '#FFA500', '#FFFF00', '#7FFF00', '#00FFFF', '#0000FF', '#9400D3'],
    beats: [300, 320, 280, 310, 290]
  }
];
// Function to open player.html with song details
function openPlayer(song) {
Consturl=player.html?song=${encodeURIComponent(song.src)}&colors=${encodeURICompone
nt(song.colors.join(','))}&beats=${encodeURIComponent(song.beats.join(','))}`;
  window.open(url, "_blank");
}
// Shuffle and play a random song
function shuffleAndPlay() {
  const randomIndex = Math.floor(Math.random() * songs.length);
```

```
  const randomSong = songs[randomIndex];
  openPlayer(randomSong);
}
// Setup Playlist and Shuffle Button on page load
window.onload = function() {
  const shuffleButton = document.createElement("button");
  shuffleButton.textContent = "□ Shuffle All";
  shuffleButton.onclick = shuffleAndPlay;
  document.body.appendChild(shuffleButton);
  const playlist = document.createElement("div");
  songs.forEach(song => {
    const songLink = document.createElement("a");
    songLink.href = "#";
    songLink.textContent = song.name;
    songLink.style.display = "block";
    songLink.onclick = () => openPlayer(song);
    playlist.appendChild(songLink);
  });
  document.body.appendChild(playlist);
};
```

(Player.html):

```
// player.html JavaScript

// Fetch song parameters from URL

const urlParams = new URLSearchParams(window.location.search);

const songSrc = urlParams.get('song');

const colors = urlParams.get('colors') ? urlParams.get('colors').split(',') : ['#000'];

const beats = urlParams.get('beats') ? urlParams.get('beats').split(',').map(Number) : [200];

const player = document.getElementById('audio-player');

const titleElement = document.getElementById('song-title');

const loopBtn = document.getElementById("loop-btn");

const loopCount = document.getElementById("loop-count");

let colorInterval;

let currentIndex = 0;

// Load and play the selected song

function playAudio() {

  player.src = songSrc;

  player.load();

  player.play().then(() => {

  startColorChange();

  }).catch(error => console.error("Playback error:", error));

}

// Change background colors dynamically based on beats
```

```
function startColorChange() {
  clearTimeout(colorInterval);
  function changeColor() {
    let randomIndex = Math.floor(Math.random() * colors.length);
    document.body.style.backgroundColor = colors[randomIndex];
    let nextBeat = beats[currentIndex % beats.length];
    currentIndex++;
    colorInterval = setTimeout(changeColor, nextBeat);
  }
  changeColor();
}
// Toggle loop mode
function toggleLoop() {
player.loop = !player.loop;
  loopBtn.classList.toggle("loop-active", player.loop);
}
// Back to Playlist
function goBackToMain() {
  window.open("index.html", "_self");
  window.close();
}
// Event Listeners
```

```
player.onplay = startColorChange;

player.onpause = () => clearTimeout(colorInterval);

player.onended = () => clearTimeout(colorInterval);

// Play song immediately on page load

playAudio();
```

TEST SCENARIO & VALIDATION:

TEST CASE ID	TEST CASE SCENARIO	INPUT/ACTION	EXPECTED RESULT	STATUS
1	Open playlist Page	Open index.html	Playlist page with song list and shuffle button is displayed	☑Pass
2	Play a Song	Click on any Song	New tab opens with player.html and selected song plays	☑Pass
3	Shffle Songs	Click Shuffle Button	Random song opens and play with visual effects	☑Pass
4	Toggle Loop Button	Click Loop Button	Current song repeats;loop button starts spinning	☑Pass
5	Toggle Light/Dark Theme	Click Theme Toggle Button	Background and text colour change according to theme	☑Pass
6	Back to playlist	Click back button on player page	User is redirected back to the playlist page(index.html)	☑Pass
7	Dynamic Background Colour	Play Song	Background changes dynamically according to beats	☑Pass

5. RESULT:

The Sound Scapes application was successfully developed and deployed using HTML5, CSS3, and JavaScript.
It fulfills all the intended objectives, providing users with:

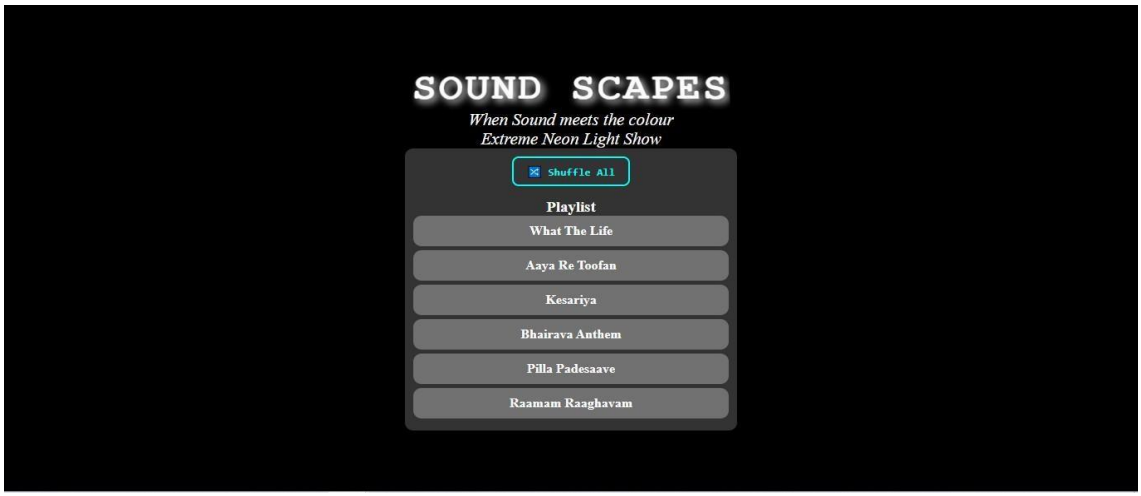
A dynamic and responsive playlist page (index.html) with shuffle functionality and a list of songs.

A vibrant player page (player.html) that:

- Plays the selected song.
- Displays animated background color transitions based on the song beats.
- Offers user interactions like Play/Pause, Loop, Theme Toggle, and Back Navigation to the playlist.
- Smooth navigation and enhanced user experience without requiring backend support or external plugins.

The system ensures that both visual and audio elements are synchronized, resulting in an engaging and immersive music experience directly within the browser.

Output Screens:



∴ Playlist Page(index.html)

∴ Player Page(player.html[Toggle theme(white)]

∴ Player Page(player.html[Toggle theme(Black)]

6. CONCLUSION:

The development of the Sound Scapes application successfully demonstrates the integration of audio playback with dynamic visual enhancements using basic web technologies such as HTML5, CSS3, and JavaScript. By combining auditory and visual elements, the project creates an engaging user experience where background colors transition rhythmically based on the beat patterns of the music being played. This approach not only makes listening to music more immersive but also adds a unique artistic layer to the standard functionality of a web-based music player.

Throughout the implementation, careful attention was given to ensuring smooth user interaction and responsiveness. Features like shuffle play, recently played tracking, looping functionality, and theme toggling were implemented to enhance usability and give users greater control over their listening experience. The project remains lightweight, requiring no server-side processing or heavy external libraries, thus ensuring fast performance across various devices and browsers.

In conclusion, Sound Scapes successfully fulfills its intended objectives and serves as a strong example of how creative UI/UX design combined with simple front-end programming can greatly enhance user engagement. Future improvements could involve adding real-time beat detection, expanding the playlist dynamically, or introducing more complex visualizations like waveforms or particle effects to further enrich the experience.

7. REFERENCE:

- MDN Web Docs - HTML5 Audio API Documentation. *(*https://developer.mozilla.org/en-US/docs/Web/HTML/Element/audio*)*
- MDN Web Docs - CSS3 Animations and Transitions. *(*https://developer.mozilla.org/enUS/docs/Web/CSS/CSS_Animations/Using_CSS_animations*)*
- MDN Web Docs - JavaScript Event Handling and DOM Manipulation. *(*https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener*)*
- W3Schools - HTML, CSS, and JavaScript Tutorials. *(*https://www.w3schools.com/*)*
- Stack Overflow - Community Discussions for JavaScript and Web Development. *(*https://stackoverflow.com/*)*
- UX Design Principles - Enhancing User Experience Through Visual Feedback. *(*https://uxdesign.cc/*)*
- FreeCodeCamp - JavaScript Projects and Practice Examples. *(*https://www.freecodecamp.org/*)*

.