# MUSIC PLAYER USING JAVASCRIPT

# A MINI PROJECTREPORT

#### Submittedby

#### Group/Team No: G6-A/T1

**JANNAT,2310990444**

**JASHAN,2310990445**

**JASLEEN,2310990446**

#### in partial fulfillment for the award of the degree of

## BACHELEOR OF ENGINEERING

***in***

COMPUTER SCIENCE & ENGINEERING

****

**CHITKARA UNIVERSITY**

**CHANDIGARH-PATIALA NATIONAL HIGHWAY**

**RAJPURA (PATIALA) PUNJAB-140401 (INDIA)**

##### OCTOBER,2023

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **SR.NO** | **CONTENTS** | **PAGE.NO** |
| 1 | Acknowledgement | 3 |
| 2 | Declaration | 4 |
| 3 | Abstract | 5 |
| 4 | Introduction | 6 |
| 5 | Problem Statement | 7 |
| 6 | Motivation | 8 |
| 7 | Approach | 10 |
| 8 | Technical Details | 11 |
| 9 | Key features | 13 |
| 10 | System architecture and Design | 14 |
| 11 | System Overview | 16 |
| 12 | Existing system | 17 |
| 13 | Project advantages | 18 |
| 14 | Source Code For Login Page   1. HTML 2. CSS 3. Output | 19  21  26 |
| 15 | Source Code For Home Page   1. HTML 2. CSS 3. JavaScript 4. Output | 27  30  34  37 |
| 16 | Source Code For About Us Page   1. HTML 2. CSS 3. Output | 38  40  45 |
| 17 | Result | 46 |
| 18 | Conclusion | 47 |
| 19 | Future scope | 48 |
| 20 | Bibliography | 49 |

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of the project would be incomplete without the emotion of the people who made it possible. First of all, we would like to thank the supreme power the Almighty God who is obviously the one has always guided us to work on the right path of life. Without his grace this project could not become a reality. Next to him are our parents, whom we are greatly indebted for us brought up with love and encouragement to this stage. We are feeling oblige in taking the opportunity to sincerely thanks to our MR. Neeraj Singla Sir. Moreover, We are highly obliged in taking the opportunity to sincerely thanks to all the staff members of computer department for their generous attitude and friends who have always helping and encouraging us though out the year. We have no valuable words to express our thanks everyone for the valuable advice and suggestions for the corrections, modification and improvement did enhance the perfection in performing our job well.

**DECLARATION**

I JASHAN BANSAl, along with my group members JANNAT ARORAand JASLEEN KAUR, students of CHITKARA UNIVERSITY , RAJPURA , PUNJAB. Here by solemnly declare that the project titled “MUSIC PLAYER” is our original as all the informative facts and figures in this documentation is based on our own experience and study during our classes.

**ABSTRACT**

The continuous growing of people’s music library requires more advanced ways of computing playlists through algorithms that match tracks to the user’s preferences. Several approaches have been made to enhance the user’s listening experience; while most of them rely on the music content provided by the user, this project presents an online application that sources the audio content from publicly available resources (YouTube). A playlist generation algorithm is developed that uses only one seed track to compute a playlist of arbitrary length. For sourcing the audio content, YouTube’s track coverage is analyzed and statistics show that, in a real-life usage scenario, almost 80% of the tracks are available while the rest have rather lower popularity. The resulting application is a fully functional but feature limited online music player that can also serve as a framework for future playlist generating algorithms or other content sources.Media usage is changing rapidly these days. This process has been ignited by several technological advances, in particular, the availability of broadband internet, the World Wide Web, affordable mass storage, and high-quality media formats, such as mp3. Many music lovers have now accumulated collections of music that have reached sizes that make it hard to maintain an overview of the data by just browsing hierarchies of folders and searching by song title or album. Search methods based on song similarity offer an alternative, allowing users to abstract from manually assigned metadata, such as, frequently imprecise or incorrect, genre information. In a context where music collections grow and change rapidly, the similarity-based organization has also the advantage of providing easy navigation and retrieval of new items, even without knowing songs by name. This opens possibilities, such as sophisticated recommendations, context-aware retrieval, and discovery of new genres and tendencies

**INTRODUCTION**

This is a project named Web Based Music Player. This project is done using HTML, CSS and JAVASCRIPT. If you are searching for a simple and lightweight solution to enable music on your website, regardless of the user's internet browser, the best solution is to use a media player that is compatible with Flash as it is normally found on most browsers . The music player allows a user to play various media file formats. It can be used to play audio as well as video files. They are portable digital music players that play music as audio files, such as MP3. In addition, most of these devices allow to store video, pictures, and to receive radio and TV programs (podcasting). Earphones and external speakers are the typical output devices delivering sound to the listener. The key features in this are as follows, It contains the ability to fetch all the mp3 songs from the Local Storage. User can create his/her favourites playlist.

Android is open source code mobile phone operating system that comes out by Google. Music player in this project is application software based on Google Android. Music is one of the best ways to relieve pressure in stressful modern society life. The purpose of this project is to develop a player which can play the mainstream file format. To browse and query the storage space as well as operation of playing can be realised. Meanwhile, this software can play, pause and select songs with latest button and next button according to sets requirement as well as set up songs.

Music is an important entertainment medium. With advancement of technology , the optimisation of manual work has gained a lot of attention. Currently, there are many traditional music players that require songs to be manually selected and organised. User , have to create and update play list for each mood, which is time consuming. Some of the music players have advanced features like providing lyrics and recommending similar songs based on the singer or genre. Although sum of the key features are enjoyable for user, there is room to improve in the field of automation when it comes to music players. Selecting song automatically and organising these based on the users mood give user a better experience. This can be accomplished through the system reacting to the users emotions , saving time that would have been spent entering information manually.

**Problem statement**

Music listeners have tough time creating and segregating the play-list manuallywhen they have hundreds of songs. It is also diﬃcult to keep track of all the songs. sometimes songs that are added and never used, wasting a lot of device memory andforcing the user to ﬁnd and delete songs manually. User’s have to manually select songs every time based on interest and mood. User’s also have diﬃculty to re-organize and playing music when play-style varies. Currently in existing application, musics organized using play-list, and play-list songs cannot be modiﬁed or altered in one click. User’s have to manually change or update each song in their play-list everytime. The sequence of songs in a play-list might not be the same every time, andsongs that a user wants to listen frequently might not be given priority or might beleft out from the list. Currently, there are no applications that allows users to playsongs on-the-go without selecting songs manually or from a play-list

To create an MP3 player using the html or css programming language to play and listen to songs, MP3 files, and other digital audio files.

The player must have a simple and easy to use interface with options for various functions and a screen around the entire playlist and buttons to turn off the player. The player should be able to play any song.

It must be able to play MP3 files or other digital audio files. It should give the user the option to pause or resume the song. The user should be given basic details about the song being played.

**MOTIVATION**

As a music lover, I’ve always felt that music players should do far more thingsthan just playing songs and allowing users to create play-lists. A music playershouldbe intelligent and act according to user’s preferences. A music player should helpusers organize and play the songs automatically without putting much eﬀort intoselection and re-organization of songs. The Emotion-Based Music Player provides abetter platform to all the music listeners, and ensures automation of song selectionand periodic updating of play-lists. This helps users organize and play songs based ontheir moods. The player should also give recommendation for users to change songson-the-go. It calculates song-weight based on EMO-algorithm.

Several solutions already use intelligent playlists embedded in music players installed on computers. There are also online solutions, the most popular of which islast.fm, which acts as a personalized radio station that plays preferredmusic. On the other hand it does not allow playback of a certain track. There are also other solutions, like the genius function of iTunes or the Music Explorer; both use the user’s music collection to generate playlists. The biggest disadvantage of the latter solution is that the user can use only tracks that he/she already has on his/her PC to generate playlists. Of course this limits the power or the algorithm very much. There are already services that provide the music content (like last.fm or YouTube to name a few) so it’s a natural conclusion to try to use these service s in connection with the playlist‐generating algorithm. In order to understand the utility of such an application, just imagine the following scenario: one enjoys listening to music while working. It is not common to store music on the company’s computer so one rather has a person al mp3 player with himself during office time. If one takes enough time to prep areones playlists in order to fit ones current mood, it is a pretty decent solution. But what if new tracks appear that one might like? One first has to do serious research in order to find them and then go through buying them, downloading them to his/her mp3 player, updating the playlists it already sounds very difficult, right? Now the suggested scenario is the following: one opens a web site, types in a track that reflects ones current moo d and hits “play”. That’s it! The player chooses tracks that one likes, also plays new tracks that one did not hear before, and can go like this for hours and ho urs without repetition. One can go on with one’s work and in order to stop the music, one only has to hit stop or close the browser. The simplicity of the solution speaks for itself. The goal of this thesis is to analyze and implement an approach of building such a web‐based music player. The questions it has to find answers for are: How should the user interaction be designed to maximize the user satisfaction? Where to source the audio data from, while ensuring a maximum coverage? And finally, how to promote the application in order to attract as many users as possible? The different implementation possibilities are evaluated and the best solution is implemented. The logic behind the web‐based music player computes a 5 sequence of tracks based on their similarity. At the same time, user behavioral data is gathered that helps further releases to be even more user friendly. Another important aspect of the application is its extensibility. Modularity and code reusing are very important parameters of this application, as it acts as a version 1.0 for future releases. These future releases will be able to interact wit h the user for finding the best track video on YouTube or to determine the mu sic preferences of users and even adapt the space to the new usage statistics.

**APPROACH**

The analysis of the currently available tools to accomplish the task is one of the most important steps because the ground concepts of the application should never change, regardless of its future complexity. The several possible implementations of the web service together with the balancing of computing tasks between server and client are the first parameters that have to be defined for a solid base. Also the programming language plays a crucial part in the development process, as it is shown later. The amount of callbacks to the database in favor of less memory usage is also an important aspect that is difficult to estimate from the start. In order to allow a high flexibility while still maintaining a small dataflow, the implementation of the logic is mainly on the server. The UI responsibility is fully retained by the client side as well as servicing UI requests and only notify the server of such activity. In order to achieve the high goals that were set, th e structure of the applications important to be highly modularized to allow interchanging the modules with better, more complex implementations. It is important to determine which components are possible and also easy to mod ularize, without introducing too much communication overhead in the interfaces. It turns out that the music content related jobs can easily be modularized, as well as the DB relatedjobs and the playlist computing tasks. The core of the application only needs to handle these modules and the logging task. Also the communication with the client is modularized, making it particularly easy to implement new clients running on the same servic e or new services to servethe same client. .

**TECHNICAL DETAILS**

Required software of the developing environment -

♣ Operation system: Windows 10,Linux

♣Software：Android SDK(Software Development Kit)、ADT(Android Development Tool)

♣JDK：Java Runtime Environment virtual machine、Java Development Kit(JDK)

**Technical Feasibility:**

To design a music player which meets the basic requirements, a deep understand of JAVA language, the Android system architecture The, application of framework and other technical knowledge are needed.(framework is the core of the application, and rules that all the programmers participating in the development must abide by). Based on the related technology information and resources for Android on the market, and equipped with technical personnel of technology and the spirit of willing to learn, the technology is feasible.

To design Android mobile phone music player as long as a computer has the

Android

To design Android mobile phone music player as long as a computer has the

Android

**Economic Feasibility:**

To design Android mobile phone music player as long as a computer has the Android development and the application development of Android is free. In addition, mobile phone music player is basic needs for public. The information that which functions are necessary form all the consumers , which functions are needed for some people, and which features are seldom to use is easy to understand. And a lot of research is eliminated, thus saved the spending. Therefore, the whole process of development doesn’t need to spend any money that is economic feasibility.

**Social Feasibility:**

With the rapid development of the mobile phone market, all kinds of audio and video resources are widely circulated on the Internet. These resources seem ordinary, but have gradually become an indispensable part of people life, which derived the development of all kinds of mobile phone player. But a lot of players devoted to fancy appearance, strong function causing a lot of wasted resources to the user's mobile phone and bringing a lot of inconvenience to the user as multitasking operation is needed. Some functions are useless to ordinary people. Powerful player is a good thing, but a lot of functions are actually useless for most users. Aimed at these problems, developing multiplied audio player which owns the features of simplified functions, common play function, meeting the needs of most users, less required memory and high quality of playing music, maximizes the optimization in performance.

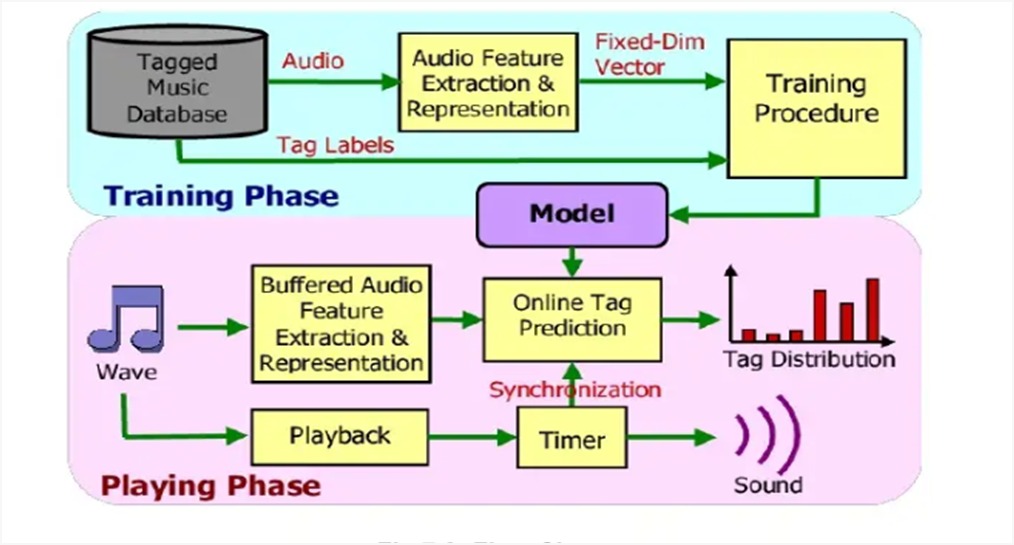
**Technologies used:**

* HTML: HTML (hypertext markup language) is a text based approached to describing how content contained with in an HTML file structure. This markup tells a web browser how to display text, images at other forms of multimedia on a web page.
* CSS:CSS stand for cascading style sheets.CSS describes how html elements are to be displayed on screen paper, or in other media. CSS saves a lot of work. It can control the layout of multiple webpages all at ones. External style sheets are stored in css files.
* JS:Java script (js) is a cross platform, object oriented programming language used by developlers to make web pages interactive. It allows developer to create dynamically updated content , use animations popup minus , clickable buttons , control multimedia extra.

.

**KEY FEATURES**

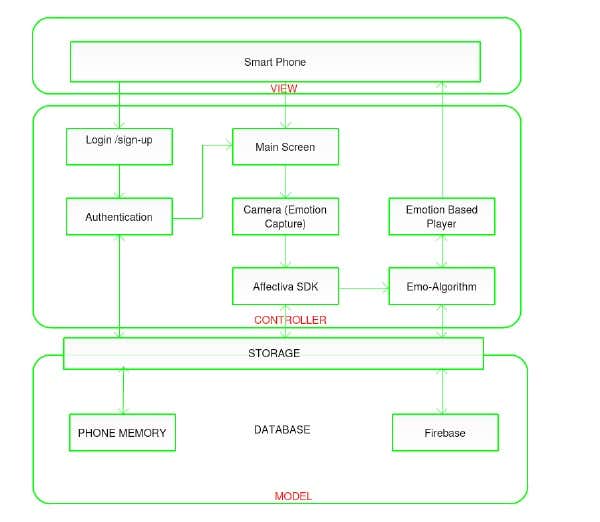
* Play
* Pause
* Stop
* Rewind
* Mute
* Add Mp3 Files
* Remove Mp3 Files



**Figure No: 1-** Diagram Of Key Features

**SYSTEM ARCHITECTURE AND DESIGN**

The system architecture of Emotion-Based Music player is shown in Fig 3.1. Theapplication is built using the architectural pattern of Model-View-Controller [13]. Itis also widely used architecture. Here, the application is divided into three mainlogical components: the model, the view and the controller.

**View**: The top layer is where the end-user communicates with the applicationthrough clicking buttons, typing details, accessing camera, selecting radio but-ton, uploading songs, etc. This layer is responsible for displaying all data or

**Figure No: 2-** Diagram Of System Architecture And Design

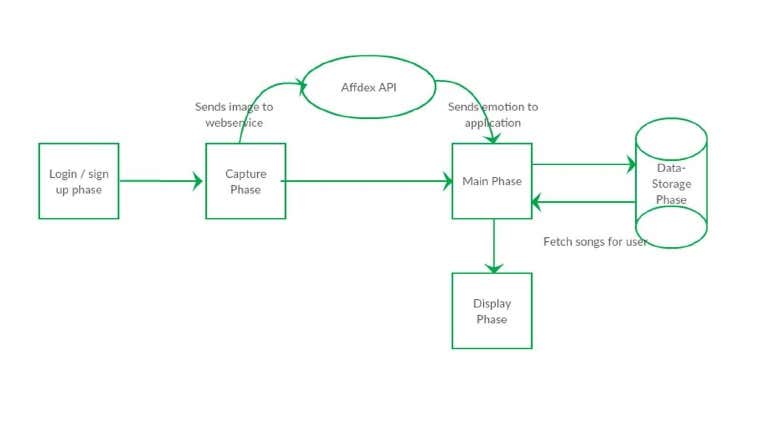
portion of data to user based on the requirement of the application. This layeralso acts as a bridge between the user and application itself. Angular 4 [4] isused in this application for displaying the output or response of the system tothe user.

•**Controller:** This middle layer of the application contains the business logic, and the main functionality of the application. As soon as the user interactswith the application, the response is processed in this layer.

 Fromlog-in todisplaying play-list, all the functions that run in background belong to thislayer. This is a mainly consists of all the functions and EMO-algorithm(discussed in later sections) which helps in segregating songs and sending output to viewlayer.

•**Model:** This layer is responsible for maintaining the user’s data. Emotion-BasedMusic Player uses Google Firebase [8] for storing user data. Firebase is veryuseful to maintain user’s proﬁles and preferences. The application also storessome temporary data on the device.A MEAN [10] stack approach is used in the development of Emotion-BasedMusic Player. Here, the MEAN stack follows MVC architecture, with every-thing written in JavaScript [9]. Both the server-side and client-side executionsare performed using JavaScript to increase the speed of the application.8

**System overview**



**Figure No: 3-** Diagram of System Overview

**Existing System**

**Currently, there are many existing music player applications . some of the inter – esting applications among them are:**

* Saavan and spotify – These applications gives good user accessibility features to play songs and recommends user with other songs of similar genre.
* Moodfuse – In this application ,user should manually enter mood and genre that wants to be heard and mood fuse recommends the songs-list.
* Steromood – User should select this mood manually by selecting the moods from the list and the application plays music from you tube.
* Musiccovery – This application has High quality songs and comprehensive music recommendations . It also suggest predefined play \_list for the user.

**Project advantages**

The benefit of using an audio player in HTML is that it allows you to add audio content to your website in a way that is easily accessible and playable for your website visitors. Here are some of the specific benefits of using an audio player in HTML**:**

1. **Better user experience:** By embedding an audio player on your website, you can provide your website visitors with an easy and intuitive way to listen to your audio content directly in their browser without having to download the audio file.
2. **Increased engagement:** Adding audio content to your website can help to increase engagement with your visitors, as audio can be more engaging and immersive than text or images alone.
3. **Versatility:** An audio player can play a variety of audio file types, such as MP3, WAV, and OGG, giving you the flexibility to use different audio formats for your content.
4. **Accessibility:** Using an audio player can also help to make your audio content more accessible to users who may have disabilities, such as visual impairments, as it allows them to listen to the audio content directly on the page.

Overall, an audio player in HTML can enhance the user experience, increase engagement, and make your audio content more accessible to a wider audience.

**Source Code For Login Page**

* **HTML:**

<!DOCTYPEhtml>

<htmllang="en">

<head>

  <metacharset="UTF-8">

  <title>Login Page in HTML with CSS Code Example</title>

  <linkhref="https://fonts.googleapis.com/css?family=Open+Sans"rel="stylesheet">

<linkhref="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"rel="stylesheet"integrity="sha384-wvfXpqpZZVQGK6TAh5PVlGOfQNHSoD2xbE+QkPxCAFlNEevoEH3Sl0sibVcOQVnN"crossorigin="anonymous"><linkrel="stylesheet"href="./login.css">

</head>

<body>

<!-- partial:index.partial.html -->

<divclass="box-form">

    <divclass="left">

        <divclass="overlay">

        <h1>Music Player</h1>

        <p>Liosten A Music And Feel It</p>

        <span>

            <p>login with social media</p>

            <ahref="https://www.facebook.com/"><iclass="fa fa-facebook"aria-hidden="true"></i></a>

            <ahref="https://twitter.com/i/flow/login?input\_flow\_data=%7B%22requested\_variant%22%3A%22eyJsYW5nIjoiZW4ifQ%3D%3D%22%7D"><iclass="fa fa-twitter"aria-hidden="true"></i> Login with Twitter</a>

        </span>

        </div>

    </div>

        <divclass="right">

        <h5>Login</h5>

        <p>Don't have an account? <ahref="#">Creat Your Account</a> it takes less than a minute</p>

        <divclass="inputs">

            <inputtype="text"placeholder="User Name">

            <br>

            <inputtype="password"placeholder="Password">

        </div>

            <br><br>

        <divclass="remember-me--forget-password">

                <!-- Angular -->

    <label>

        <inputtype="checkbox"name="item"checked/>

        <spanclass="text-checkbox">Remember me</span>

    </label>

            <p>forget password?</p>

        </div>

            <br>

            <button><ahref="music.html">Login</a></button>

    </div>

</div>

<!-- partial -->

</body>

</html>

* **CSS:**

body {

  background-image: linear-gradient(135deg, #FAB2FF 10%, #1904E5 100%);

  background-size: cover;

  background-repeat: no-repeat;

  background-attachment: fixed;

  font-family: "Open Sans", sans-serif;

  color: #333333;

}

.box-form {

  margin: 0 auto;

  width: 80%;

  background: #FFFFFF;

  border-radius: 10px;

  overflow: hidden;

  display: flex;

  flex: 11100%;

  align-items: stretch;

  justify-content: space-between;

  box-shadow: 0020px6px #090b6f85;

}

@media (max-width: 980px) {

  .box-form {

    flex-flow: wrap;

    text-align: center;

    align-content: center;

    align-items: center;

  }

}

.box-formdiv {

  height: auto;

}

.box-form.left {

  color: #FFFFFF;

  background-size: cover;

  background-repeat: no-repeat;

  background-image: url("https://i.pinimg.com/736x/5d/73/ea/5d73eaabb25e3805de1f8cdea7df4a42--tumblr-backgrounds-iphone-phone-wallpapers-iphone-wallaper-tumblr.jpg");

  overflow: hidden;

}

.box-form.left.overlay {

  padding: 30px;

  width: 100%;

  height: 100%;

  background: #5961f9ad;

  overflow: hidden;

  box-sizing: border-box;

}

.box-form.left.overlayh1 {

  font-size: 10vmax;

  line-height: 1;

  font-weight: 900;

  margin-top: 40px;

  margin-bottom: 20px;

}

.box-form.left.overlayspanp {

  margin-top: 30px;

  font-weight: 900;

}

.box-form.left.overlayspana {

  background: #3b5998;

  color: #FFFFFF;

  margin-top: 10px;

  padding: 14px50px;

  border-radius: 100px;

  display: inline-block;

  box-shadow: 03px6px1px #042d4657;

}

.box-form.left.overlayspana:last-child {

  background: #1dcaff;

  margin-left: 30px;

}

.box-form.right {

  padding: 40px;

  overflow: hidden;

}

@media (max-width: 980px) {

  .box-form.right {

    width: 100%;

  }

}

.box-form.righth5 {

  font-size: 6vmax;

  line-height: 0;

}

.box-form.rightp {

  font-size: 14px;

  color: #B0B3B9;

}

.box-form.right.inputs {

  overflow: hidden;

}

.box-form.rightinput {

  width: 100%;

  padding: 10px;

  margin-top: 25px;

  font-size: 16px;

  border: none;

  outline: none;

  border-bottom: 2px solid #B0B3B9;

}

.box-form.right.remember-me--forget-password {

  display: flex;

  justify-content: space-between;

  align-items: center;

}

.box-form.right.remember-me--forget-passwordinput {

  margin: 0;

  margin-right: 7px;

  width: auto;

}

.box-form.rightbutton {

  float: right;

  color: #fff;

  font-size: 16px;

  padding: 12px35px;

  border-radius: 50px;

  display: inline-block;

  border: 0;

  outline: 0;

  box-shadow: 0px4px20px0px #49c628a6;

  background-image: linear-gradient(135deg, #70F570 10%, #49C628 100%);

}

label {

  display: block;

  position: relative;

  margin-left: 30px;

}

label::before {

  content: ' \f00c';

  position: absolute;

  font-family: FontAwesome;

  background: transparent;

  border: 3px solid #70F570;

  border-radius: 4px;

  color: transparent;

  left: -30px;

  transition: all 0.2s linear;

}

label:hover::before {

  font-family: FontAwesome;

  content: ' \f00c';

  color: #fff;

  cursor: pointer;

  background: #70F570;

}

label:hover::before.text-checkbox {

  background: #70F570;

}

labelspan.text-checkbox {

  display: inline-block;

  height: auto;

  position: relative;

  cursor: pointer;

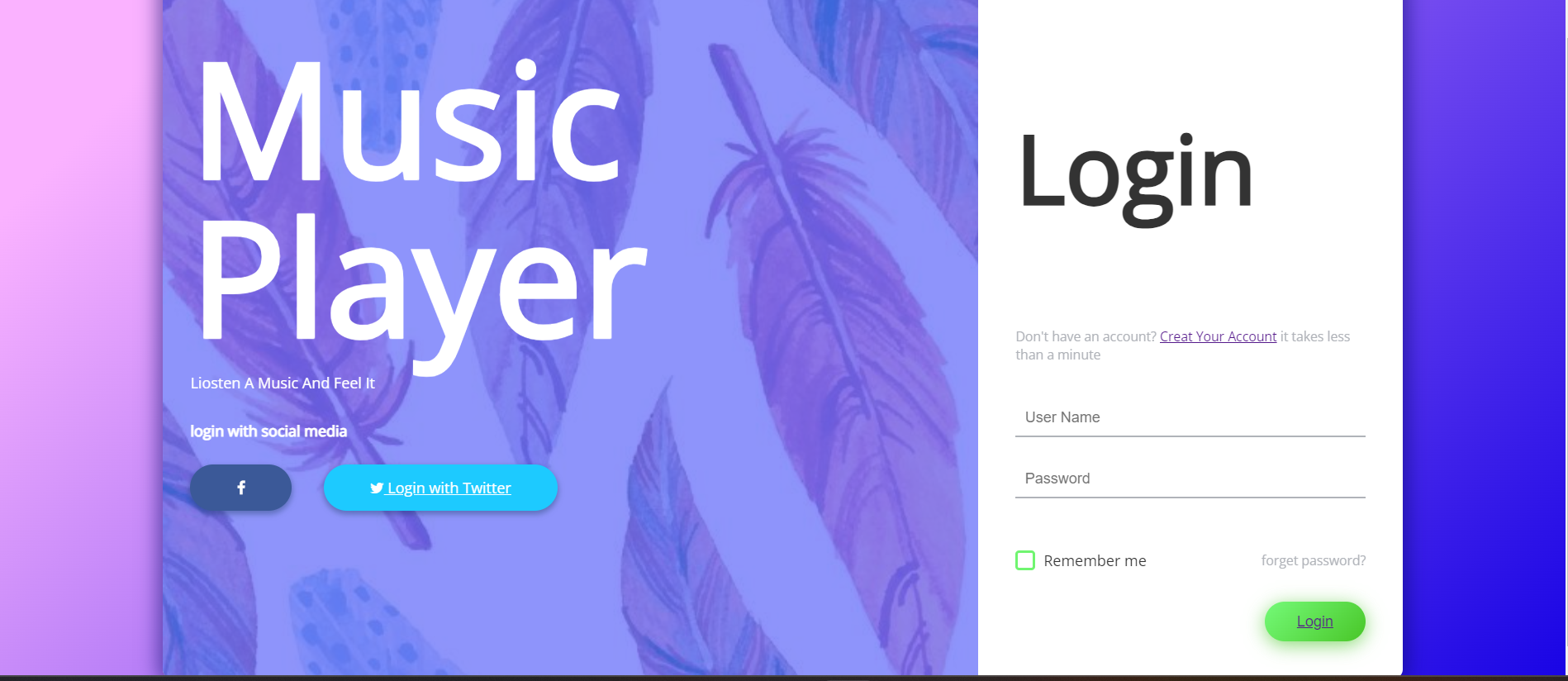
  transition: all 0.2s linear;

}

labelinput[type="checkbox"] {

  display: none;

}

**OUTPUT:**

**Figure No: 4-** Output For Login Page

**Source Code For Home Page**

* **HTML:**

<!DOCTYPEhtml>

<htmllang="en">

    <head>

        <metacharset="UTF-8">

        <m etahttp-equiv="X-UA-Compatible"content="IE=edge">

        <metaname="viewport"content="width=device-width, initial-scale=1.0">

        <title>Spotify - Your favourite music is here</title>

        <linkrel="stylesheet"href="music.css">

        </head>

<body>

    <nav>

        <ul>

            <liclass="brand"><imgsrc="logo.png"alt="Spotify"> Spotify</li>

            <li><ahref="">Home</a></li>

            <li><ahref="about.html">About Us</a></li>

            <li><ahref="login.html">Login</a></li>

        </ul>

    </nav>

    <divclass="container">

        <divclass="songList">

            <h1>Best of NCS - No Copyright Sounds</h1>

            <divclass="songItemContainer">

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Feelings</span>

                    <spanclass="songlistplay"><spanclass="timestamp">3:44<iid="0"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Pee Loon</span>

                    <spanclass="songlistplay"><spanclass="timestamp">4:47<iid="1"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Softly</span>

                    <spanclass="songlistplay"><spanclass="timestamp">2:35<iid="2"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Bachke-Bachke</span>

                    <spanclass="songlistplay"><spanclass="timestamp">3:30<iid="3"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Temporary Pyar</span>

                    <spanclass="songlistplay"><spanclass="timestamp">3:41<iid="4"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Libas</span>

                    <spanclass="songlistplay"><spanclass="timestamp">4:27 <iid="5"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Arjan Velly</span>

                    <spanclass="songlistplay"><spanclass="timestamp">3:00<iid="6"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Father Saab</span>

                    <spanclass="songlistplay"><spanclass="timestamp">4:03<iid="7"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Daru Badnaam</span>

                    <spanclass="songlistplay"><spanclass="timestamp">4:00<iid="8"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

                <divclass="songItem">

                    <imgalt="1">

                    <spanclass="songName">Main Rang Sharvatoon Ka</span>

                    <spanclass="songlistplay"><spanclass="timestamp">4:30<iid="9"class="far songItemPlay fa-play-circle"></i></span></span>

                </div>

            </div>

        </div>

        <divclass="songBanner"></div>

    </div>

    <divclass="bottom">

        <inputtype="range"name="range"id="myProgressBar"min="0"value="0"max="100">

        <divclass="icons">

            <!--fontawesome icons -->

            <iclass="fas fa-3x fa-step-backward"id="previous"></i>

            <iclass="far fa-3x fa-play-circle"id="masterPlay"></i>

            <iclass="fas fa-3x fa-step-forward"id="next"></i>

        </div>

        <divclass="songInfo">

            <imgsrc="playing.gif"width="42px"alt=""id="gif"><spanid="masterSongName">Feelings</span>

        </div>

    </div>

    <scriptsrc="music.js"></script>

    <scriptsrc="https://kit.fontawesome.com/26504e4a1f.js"crossorigin="anonymous"></script>

</body>

</html>

* **CSS:**

@import url('https://fonts.googleapis.com/css2?family=Varela+Round&display=swap');

body{

    background-color: antiquewhite;

}

\*{

    margin: 0;

    padding: 0;

}

nav{

    font-family: 'Ubuntu', sans-serif;

}

navul{

    display: flex;

    align-items: center;

    list-style-type: none;

    height: 65px;

    background-color: black;

    color: white;

}

navulli{

    padding: 012px;

    cursor: pointer;

}

.brandimg{

    width: 44px;

    padding: 08px;

}

.brand {

    display: flex;

    align-items: center;

    font-weight: bolder;

    font-size: 1.3rem;

}

.container{

    min-height: 72vh;

    background-color: black;

    color: white;

   font-family: 'Varela Round', sans-serif;

   display: flex;

   margin: 23px auto;

   width: 80%;

   border-radius: 12px;

   padding: 34px;

   background-image: url('bg.jpg');

}

.bottom{

    position: sticky;

    bottom: 0;

    height: 135px;

    background-color: black;

    color: white;

    display: flex;

    justify-content: center;

    align-items: center;

    flex-direction: column;

}

.icons{

    margin-top: 14px;

}

.iconsi{

    cursor: pointer;

}

#myProgressBar{

    width: 80vw;

    cursor: pointer;

}

.songItemContainer{

    margin-top: 74px;

}

.songItem{

    height: 50px;

    display: flex;

    background-color: white;

    color: black;

    margin: 12px0;

    justify-content: space-between;

    align-items: center;

    border-radius: 34px;

}

.songItemimg{

    width: 43px;

    margin: 023px;

    border-radius: 34px;

}

.timestamp{

    margin: 023px;

}

.timestampi{

    cursor: pointer;

}

.songInfo{

    position: absolute;

    left: 10vw;

    font-family: 'Varela Round', sans-serif;

}

.songInfoimg{

    opacity: 0;

    transition: opacity 0.4s ease-in;

}

@media only screen and (max-width: 1100px) {

    body {

      background-color: red;

    }

  }

* **JavaScript:**

console.log("Welcome to Spotify");

// Initialize the Variables

letsongIndex = 0;

letaudioElement = newAudio('songs/1.mp3');

letmasterPlay = document.getElementById('masterPlay');

letmyProgressBar = document.getElementById('myProgressBar');

letgif = document.getElementById('gif');

letmasterSongName = document.getElementById('masterSongName');

letsongItems = Array.from(document.getElementsByClassName('songItem'));

letsongs = [

    {songName: "Feelings", filePath: "songs/1.mp3", coverPath: "covers/1.jpg"},

    {songName: "Pee Loon", filePath: "songs/2.mp3", coverPath: "covers/2.jpg"},

    {songName: "Softly", filePath: "songs/3.mp3", coverPath: "covers/3.jpg"},

    {songName: "Bachke-Bachke", filePath: "songs/4.mp3", coverPath: "covers/4.jpg"},

    {songName: "Temporary Pyar", filePath: "songs/5.mp3", coverPath: "covers/5.jpg"},

    {songName: "Libas", filePath: "songs/2.mp3", coverPath: "covers/6.jpg"},

    {songName: "Arjan Vailly", filePath: "songs/2.mp3", coverPath: "covers/7.jpg"},

    {songName: "Father Saab", filePath: "songs/2.mp3", coverPath: "covers/8.jpg"},

    {songName: "Daru Badnaam", filePath: "songs/2.mp3", coverPath: "covers/9.jpg"},

    {songName: "Main Rang Sharbatoon Ka", filePath: "songs/4.mp3", coverPath: "covers/10.jpg"},

]

songItems.forEach((element, i)=>{

    element.getElementsByTagName("img")[0].src = songs[i].coverPath;

    element.getElementsByClassName("songName")[0].innerText = songs[i].songName;

})

// Handle play/pause click

masterPlay.addEventListener('click', ()=>{

    if(audioElement.paused ||audioElement.currentTime<=0){

        audioElement.play();

        masterPlay.classList.remove('fa-play-circle');

        masterPlay.classList.add('fa-pause-circle');

        gif.style.opacity = 1;

    }

    else{

        audioElement.pause();

        masterPlay.classList.remove('fa-pause-circle');

        masterPlay.classList.add('fa-play-circle');

        gif.style.opacity = 0;

    }

})

// Listen to Events

audioElement.addEventListener('timeupdate', ()=>{

    // Update Seekbar

    progress = parseInt((audioElement.currentTime/audioElement.duration)\*100);

    myProgressBar.value = progress;

})

myProgressBar.addEventListener('change', ()=>{

    audioElement.currentTime =myProgressBar.value \*audioElement.duration/100;

})

constmakeAllPlays = ()=>{

    Array.from(document.getElementsByClassName('songItemPlay')).forEach((element)=>{

        element.classList.remove('fa-pause-circle');

        element.classList.add('fa-play-circle');

    })

}

Array.from(document.getElementsByClassName('songItemPlay')).forEach((element)=>{

    element.addEventListener('click', (e)=>{

        makeAllPlays();

        songIndex = parseInt(e.target.id);

        e.target.classList.remove('fa-play-circle');

        e.target.classList.add('fa-pause-circle');

        audioElement.src = `songs/${songIndex+1}.mp3`;

        masterSongName.innerText = songs[songIndex].songName;

        audioElement.currentTime = 0;

        audioElement.play();

        gif.style.opacity = 1;

        masterPlay.classList.remove('fa-play-circle');

        masterPlay.classList.add('fa-pause-circle');

    })

})

document.getElementById('next').addEventListener('click', ()=>{

    if(songIndex>=9){

        songIndex = 0

    }

    else{

        songIndex += 1;

    }

    audioElement.src = `songs/${songIndex+1}.mp3`;

    masterSongName.innerText = songs[songIndex].songName;

    audioElement.currentTime = 0;

    audioElement.play();

    masterPlay.classList.remove('fa-play-circle');

    masterPlay.classList.add('fa-pause-circle');

})

document.getElementById('previous').addEventListener('click', ()=>{

    if(songIndex<=0){

        songIndex = 0

    }

    else{

        songIndex -= 1;

    }

    audioElement.src = `songs/${songIndex+1}.mp3`;

    masterSongName.innerText = songs[songIndex].songName;

    audioElement.currentTime = 0;

    audioElement.play();

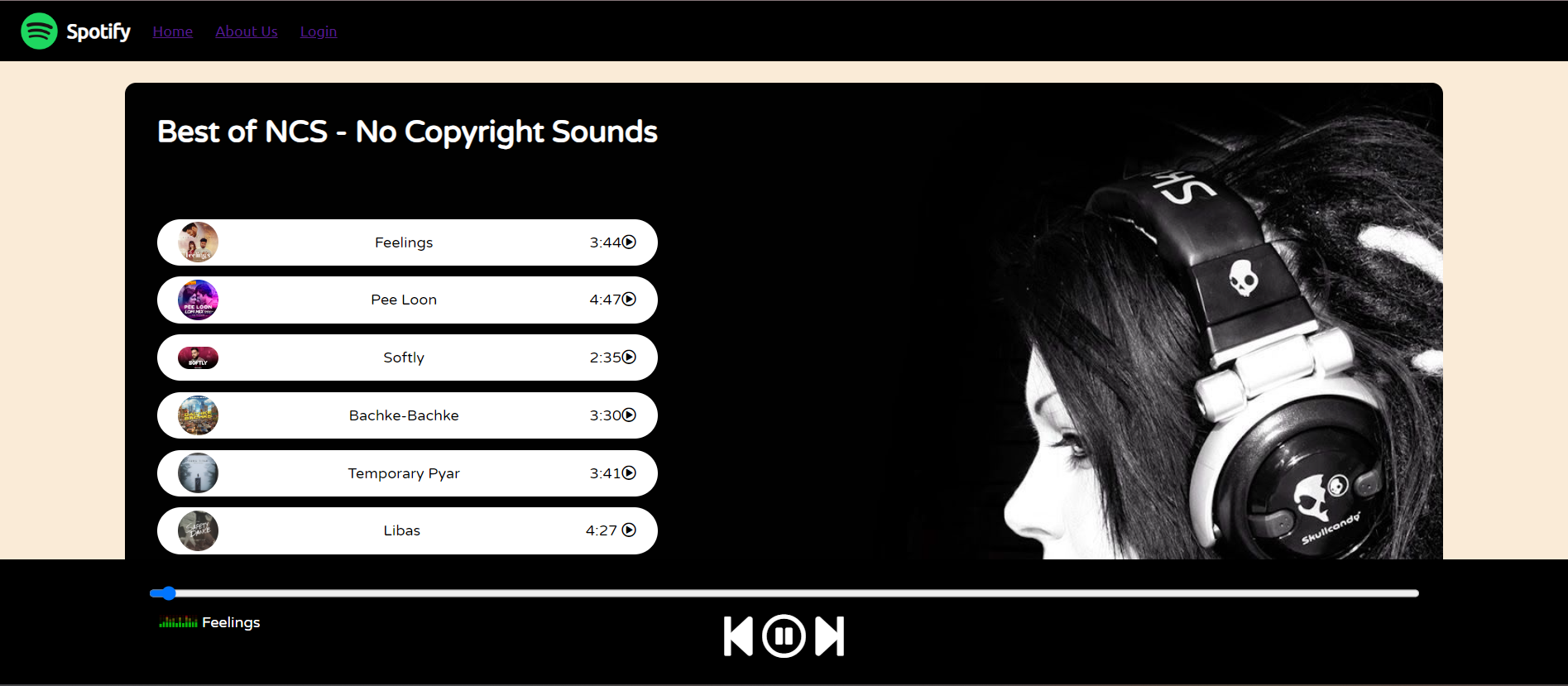
    masterPlay.classList.remove('fa-play-circle');

    masterPlay.classList.add('fa-pause-circle');

})

**Output:**

**Figure No: 5-** Output For Home Page

****

**Source Code For About Us Page**

* **HTML:**

<!DOCTYPEhtml>

<htmllang="en">

<head>

    <metacharset="UTF-8">

    <metaname="viewport"content="width=device-width, initial-scale=1.0">

    <linkrel="stylesheet"href="about.css">

    <title>Document</title>

</head>

<body>

    <divclass="responsive-container-block bigContainer">

        <divclass="responsive-container-block Container">

          <pclass="text-blk heading">

            About Us

          </p>

          <pclass="text-blksubHeading">

            We provide numerous Spotify Service options. Certain Spotify Service options are provided free-of-charge, while other options require payment before they can be accessed (the "Paid Subscriptions"). We may also offer special promotional plans, memberships, or services, including offerings of third-party products and services. We are not responsible for the products and services provided by such third parties.The Unlimited Service may not be available to all users. We will explain which services are available to you when you are signing up for the services. If you cancel your subscription to the Unlimited Service, or if your subscription to the Unlimited Service is interrupted (for example, if you change your payment details), you may not be able to re-subscribe for the Unlimited Service. Note that the Unlimited Service may be discontinued in the future, in which case you will no longer be charged for the Service.

          </p>

          <divclass="social-icons-container">

            <aclass="social-icon"href="https://www.facebook.com/">

              <imgclass="socialIcon image-block"src="https://workik-widget-assets.s3.amazonaws.com/widget-assets/images/bb33.png">

            </a>

            <aclass="social-icon"href="https://www.linkedin.com/feed/">

              <imgclass="socialIcon image-block"src="https://workik-widget-assets.s3.amazonaws.com/widget-assets/images/bb34.png">

            </a>

            <aclass="social-icon"href="https://www.instagram.com/">

              <imgclass="socialIcon image-block"src="https://workik-widget-assets.s3.amazonaws.com/widget-assets/images/bb35.png">

            </a>

            <aclass="social-icon"href="https://twitter.com/i/flow/login?input\_flow\_data=%7B%22requested\_variant%22%3A%22eyJsYW5nIjoiZW4ifQ%3D%3D%22%7D">

              <imgclass="socialIcon image-block"src="https://workik-widget-assets.s3.amazonaws.com/widget-assets/images/bb36.png">

            </a>

          </div>

        </div>

      </div>

      <scriptsrc="about.js"></script>

</body>

</html>

* **CSS:**

\* {

    font-family: Nunito, sans-serif;

  }

  .text-blk {

    margin-top: 0px;

    margin-right: 0px;

    margin-bottom: 0px;

    margin-left: 0px;

    padding-top: 0px;

    padding-right: 0px;

    padding-bottom: 0px;

    padding-left: 0px;

    line-height: 25px;

  }

  .responsive-container-block {

    min-height: 75px;

    height: fit-content;

    width: 100%;

    padding-top: 10px;

    padding-right: 10px;

    padding-bottom: 10px;

    padding-left: 10px;

    display: flex;

    flex-wrap: wrap;

    margin-top: 0px;

    margin-right: auto;

    margin-bottom: 0px;

    margin-left: auto;

    justify-content: flex-start;

  }

  .responsive-container-block.bigContainer {

    padding-top: 10px;

    padding-right: 30px;

    padding-bottom: 10px;

    padding-left: 30px;

    background-image: url("https://workik-widget-assets.s3.amazonaws.com/widget-assets/images/bb29.png");

    background-position-x: initial;

    background-position-y: initial;

    background-size: cover;

    background-attachment: initial;

    background-origin: initial;

    background-clip: initial;

    background-color: initial;

    background-repeat-x: no-repeat;

    background-repeat-y: no-repeat;

  }

  .responsive-container-block.Container {

    max-width: 800px;

    flex-direction: column;

    align-items: center;

    padding-top: 20px;

    padding-right: 20px;

    padding-bottom: 20px;

    padding-left: 20px;

    margin-top: 150px;

    margin-right: auto;

    margin-bottom: 150px;

    margin-left: auto;

    background-color: white;

    border-top-left-radius: 10px;

    border-top-right-radius: 10px;

    border-bottom-right-radius: 10px;

    border-bottom-left-radius: 10px;

  }

  .text-blk.heading {

    font-size: 36px;

    line-height: 45px;

    font-weight: 800;

    margin-top: 0px;

    margin-right: 0px;

    margin-bottom: 30px;

    margin-left: 0px;

  }

  .text-blk.subHeading {

    text-align: center;

    font-size: 18px;

    line-height: 26px;

    margin-top: 0px;

    margin-right: 0px;

    margin-bottom: 60px;

    margin-left: 0px;

  }

  .socialIcon {

    width: 33px;

    height: 33px;

  }

  .social-icons-container {

    display: flex;

    flex-direction: row;

    align-items: flex-start;

  }

  .social-icon {

    margin: 050px050px;

  }

  .social-icon:hover {

    cursor: pointer;

  }

  @media (max-width: 768px) {

    .text-blk.heading {

      font-size: 55px;

      line-height: 65px;

    }

    .text-blk.subHeading {

      font-size: 18px;

      line-height: 24px;

    }

    .socialIcon {

      width: 20px;

      height: 20px;

    }

    .text-blk.subHeading {

      line-height: 27px;

    }

    .text-blk.heading {

      font-size: 32px;

      line-height: 40px;

      margin-top: 0px;

      margin-right: 0px;

      margin-bottom: 20px;

      margin-left: 0px;

    }

    .social-icon {

      margin: 025px025px;

    }

  }

  @media (max-width: 500px) {

    .responsive-container-block.bigContainer {

      padding-top: 10px;

      padding-right: 20px;

      padding-bottom: 10px;

      padding-left: 20px;

    }

    .text-blk.heading {

      font-size: 45px;

      line-height: 55px;

      margin-top: 0px;

      margin-right: 0px;

      margin-bottom: 20px;

      margin-left: 0px;

    }

    .text-blk.subHeading {

      font-size: 14px;

      line-height: 22px;

      margin-top: 0px;

      margin-right: 0px;

      margin-bottom: 30px;

      margin-left: 0px;

    }

    .social-icons-container {

      flex-wrap: wrap;

      align-items: center;

      justify-content: center;

    }

    .text-blk.subHeading {

      font-size: 16px;

      line-height: 23px;

    }

    .text-blk.heading {

      font-size: 26px;

      line-height: 30px;

    }

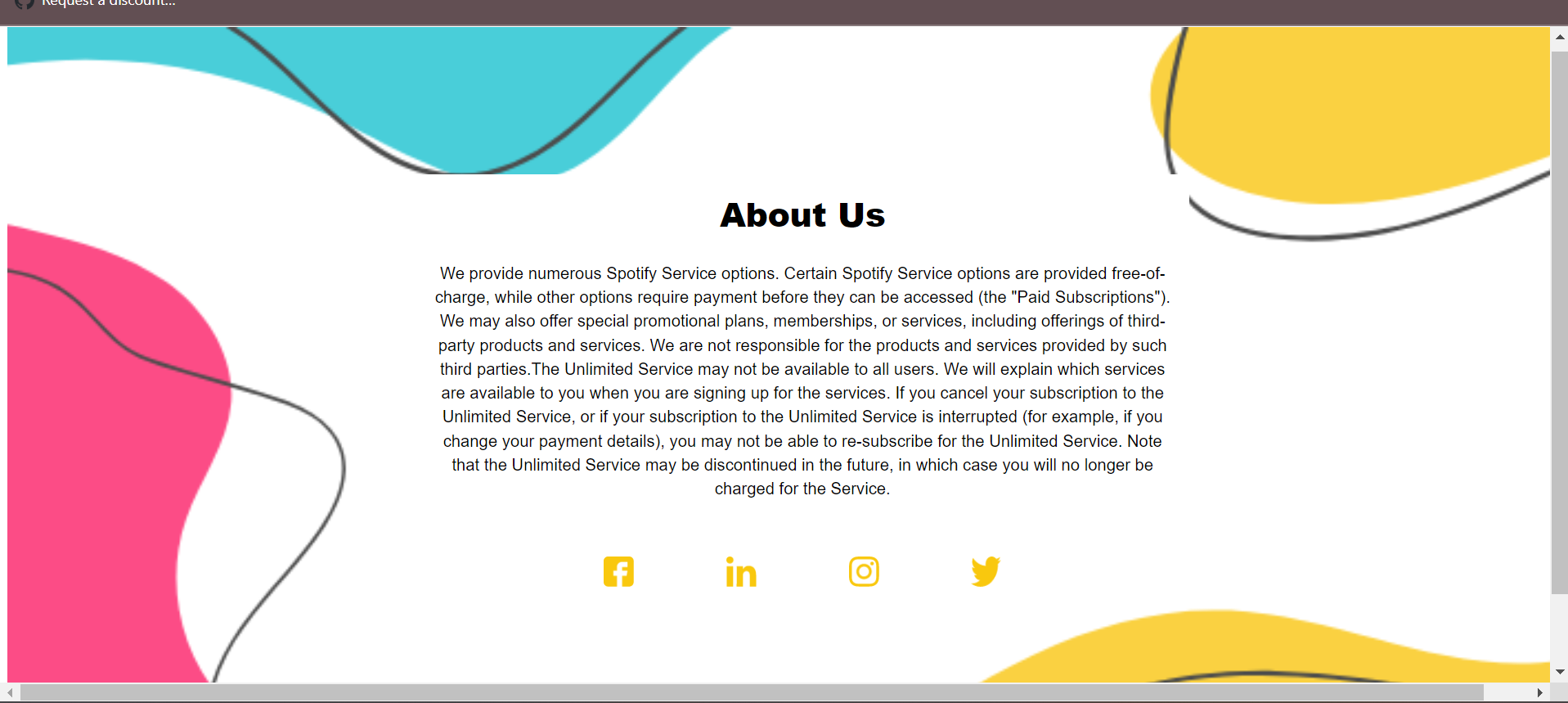
    .social-icon {

      margin: 020px020px;

    }

  }

**Output:**

****

**Figure No: 6-** Output For About Us Page

**Result**

The result presented in this project has utilised HTML -5 technology to support multiple hardware platforms. Even though it is less neither stable nor compatible , as more and more major web browsers start to support or improve the current support of the web audio API audio engine as well as animation , the future of the development capability is brighter.

**CONCLUSION**

The project named Web Based Music Player has been studied successfully above. All the requirements as well as working of the project along with the technology stack used has been explained carefully keeping each and every detail in mind. The project works completely fine and we may conclude that the project is still in its developing stage. The project can be made better with time but right now too it works satisfactorily.

**Dependencies:**

1. The web application can currently only run on browser HTML
2. The application requires a mandatory user based instruction
3. The system requires a node severStrengths:

1.Provides a better song listening experience to all users.

2.Fully automated system with single page application.

3.Easy design and implementation.

4.Implementing high level concepts such as java script and angular module.

5.Extensible.

6.Robust.

7.Can be used at any locale without much changes required.

8.Cost Effective.

9.Space Age Technology.

10.Efficient working and good results.

11.Customizable.

**Future Scope**

**The future scope of the project can be as follows:**

1.The application can be made more interactive and customizable.

2.The application can be trained more for better accuracy.

3.The application can be joined with other modules such some websites that have to use asmall part of music running in background hence they can use API of this.

4.The project can be used to teach angularjs and node js basic fundamentals of theirfunctionality

5.Current application uses Aﬀectiva SDK that has a lot of limitations, creatingcustom emotion recognition system that can be merged into the current application improves functionality and performance of the system.

6.Making the application run without needing an internet connection.

7.Including other emotions

8.Playing songs automatically

9.Optimizing the EMO-algorithm by including additional features which helpssystem to categorize user based on many other factors like location and sug-gesting the user to travel to that location and play songs accordingly

**BIBLIOGRAPHY**

1)YOUTUBE

2)GOOGLE

3)CHAT GPT

4)BOOKS