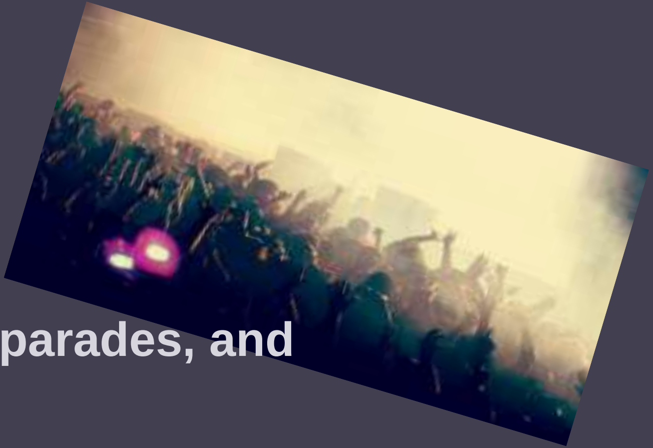
Project: *Music Player* *Presentation*

Binding People Together

- **Concerts, clubs, parties, etc.** allow people to bond and experience music together



- Music plays an important role in special occasions such as **weddings, parades, and religious festivities**

- "…means of giving voice to their common experience and to give vent to anger, grief and frustration at their mistreatment."

Evoke Strong Emotions/Feelings:

- Motivation
- Sadness
- Euphoria

About Project:

This project [Music Player in JavaScript] has several features like you can loop, repeat or shuffle a song, play/pause a song or play the next or previous song. You can view your songs list and also know which song is currently playing and you can also select the song from the list to play.

About The Code~

To create this project [Custom Music Player in JavaScript]. First, we created four files, HTML File, CSS File, and JavaScript File.

First, we created an HTML file with the name of index.html and paste the given codes in your HTML file. Remember, you've to create a file with .html extension and the images that are used on this music player won't appear.

Code Continues...

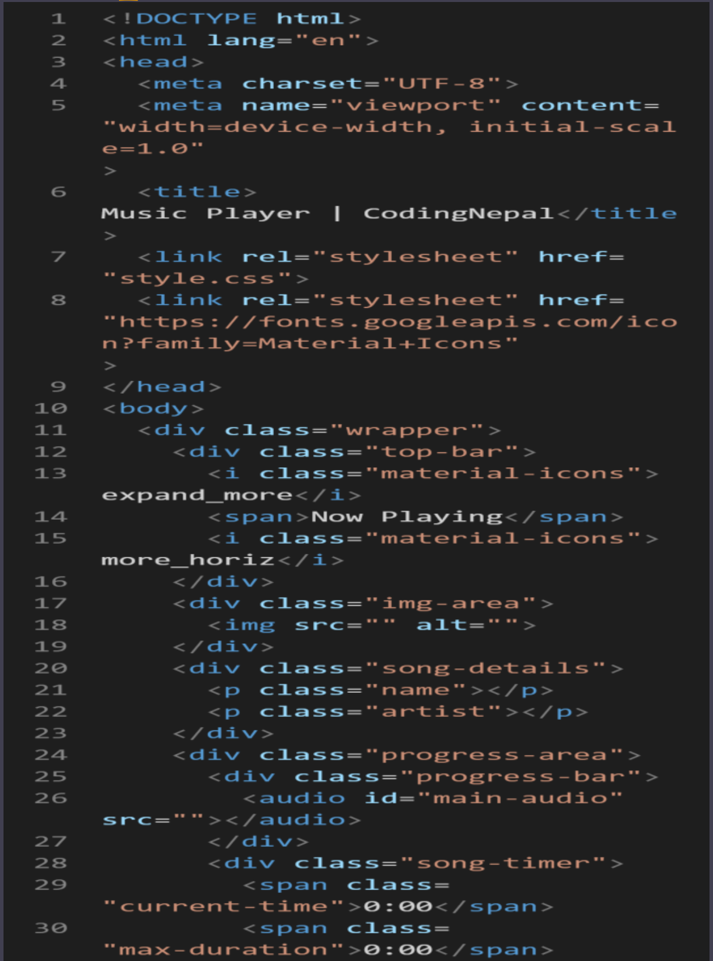
Second, we created a CSS file with the name of style.css and paste the given codes in your CSS file.

Then Third, a JavaScript file with the name of music-list.js and paste the given codes in your JavaScript file. In this JavaScript file, we store all music details using the array.

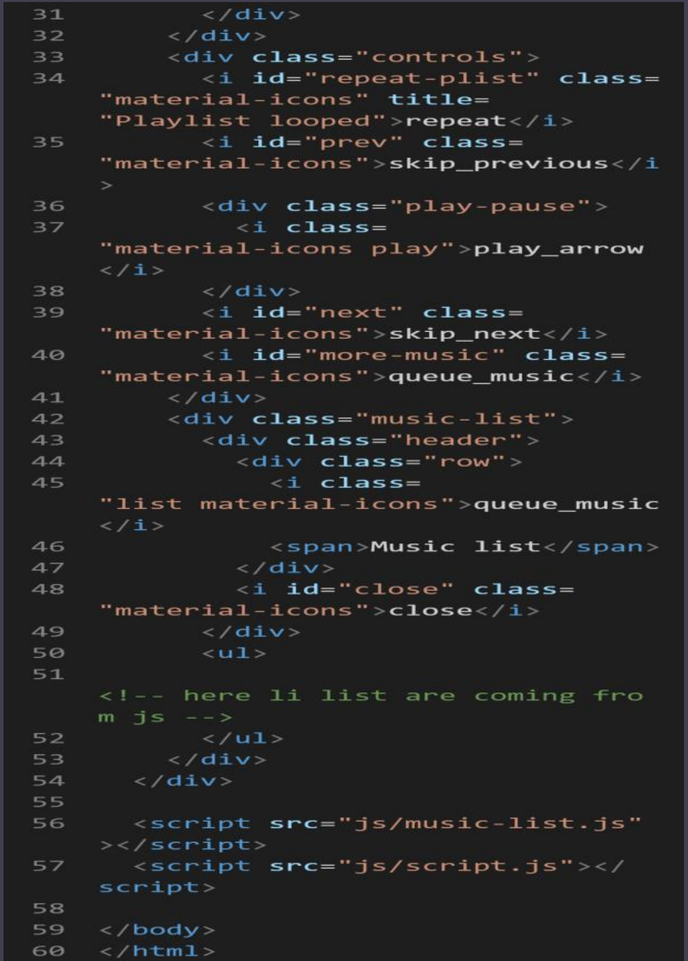
~And Last

Last, create a JavaScript file with the name of script.js and paste the given codes in your JavaScript file.

Input:html

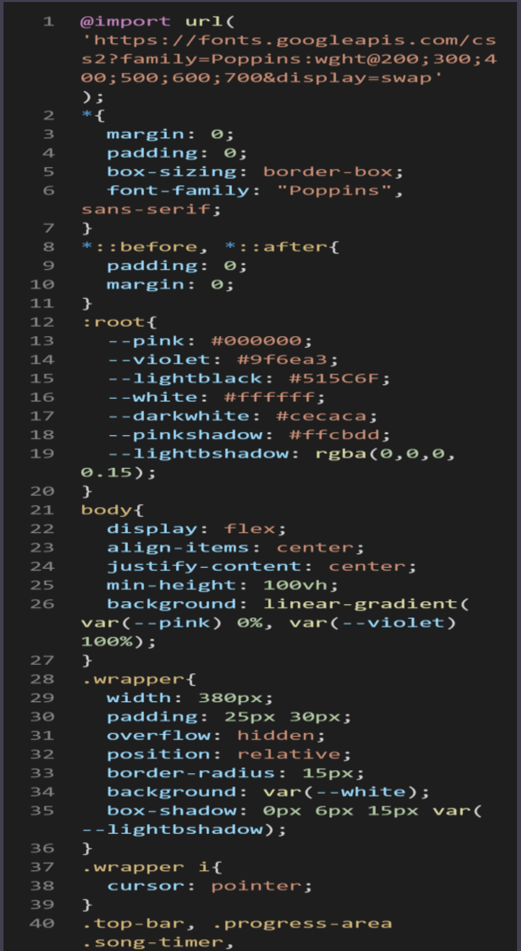


```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Music Player | CodingNepal</title>
  <link rel="stylesheet" href="style.css">
  <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
</head>
<body>
  <div class="wrapper">
    <div class="top-bar">
      <i class="material-icons">expand_more</i>
      <span>Now Playing</span>
      <i class="material-icons">more_horiz</i>
    </div>
    <div class="img-area">
      <img src="" alt="">
    </div>
    <div class="song-details">
      <p class="name"></p>
      <p class="artist"></p>
    </div>
    <div class="progress-area">
      <div class="progress-bar">
        <audio id="main-audio" src=""></audio>
      </div>
      <div class="song-timer">
        <span class="current-time">0:00</span>
        <span class="max-duration">0:00</span>
```

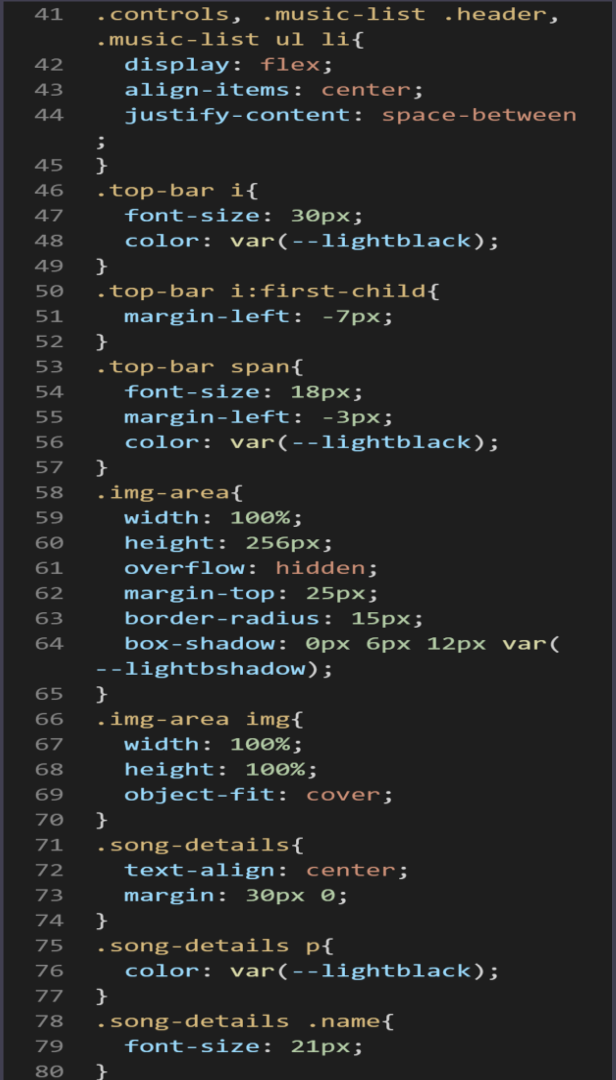


```html
      </div>
    </div>
    <div class="controls">
      <i id="repeat-plist" class="material-icons" title="Playlist looped">repeat</i>
      <i id="prev" class="material-icons">skip_previous</i>
      <div class="play-pause">
        <i class="material-icons play">play_arrow</i>
      </div>
      <i id="next" class="material-icons">skip_next</i>
      <i id="more-music" class="material-icons">queue_music</i>
    </div>
    <div class="music-list">
      <div class="header">
        <div class="row">
          <i class="list material-icons">queue_music</i>
          <span>Music list</span>
        </div>
        <i id="close" class="material-icons">close</i>
      </div>
      <ul>
        <!-- here li list are coming from js -->
      </ul>
    </div>
  </div>

  <script src="js/music-list.js"></script>
  <script src="js/script.js"></script>

</body>
</html>
```

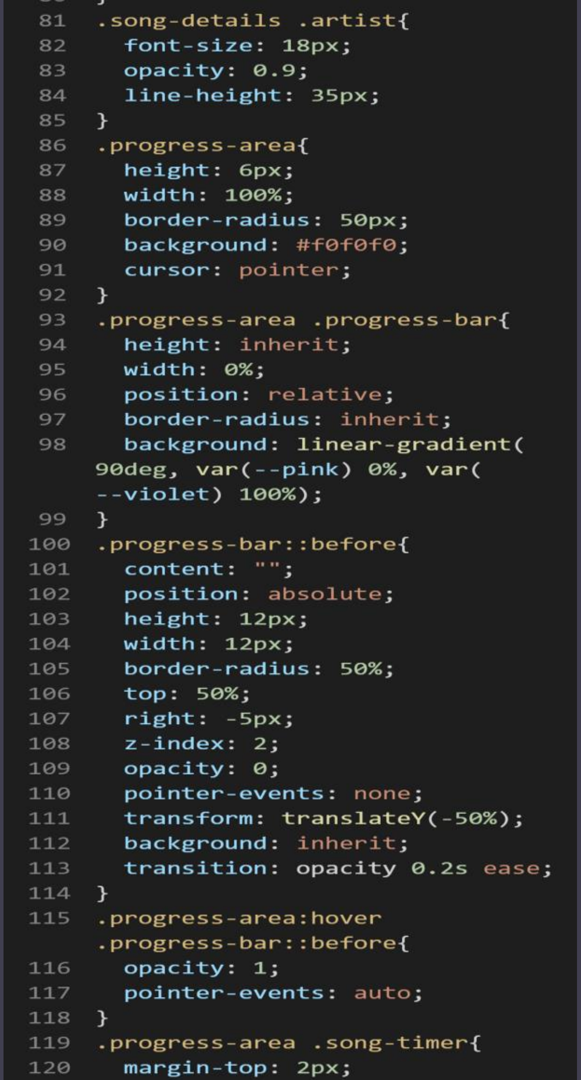
Input:css



```css
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap');
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}
*::before, *::after{
  padding: 0;
  margin: 0;
}
:root{
  --pink: #000000;
  --violet: #9f6ea3;
  --lightblack: #515C6F;
  --white: #ffffff;
  --darkwhite: #cecaca;
  --pinkshadow: #ffcbdd;
  --lightbshadow: rgba(0,0,0,0.15);
}
body{
  display: flex;
  align-items: center;
  justify-content: center;
  min-height: 100vh;
  background: linear-gradient(var(--pink) 0%, var(--violet) 100%);
}
.wrapper{
  width: 380px;
  padding: 25px 30px;
  overflow: hidden;
  position: relative;
  border-radius: 15px;
  background: var(--white);
  box-shadow: 0px 6px 15px var(--lightbshadow);
}
.wrapper i{
  cursor: pointer;
}
.top-bar, .progress-area
.song-timer,
```

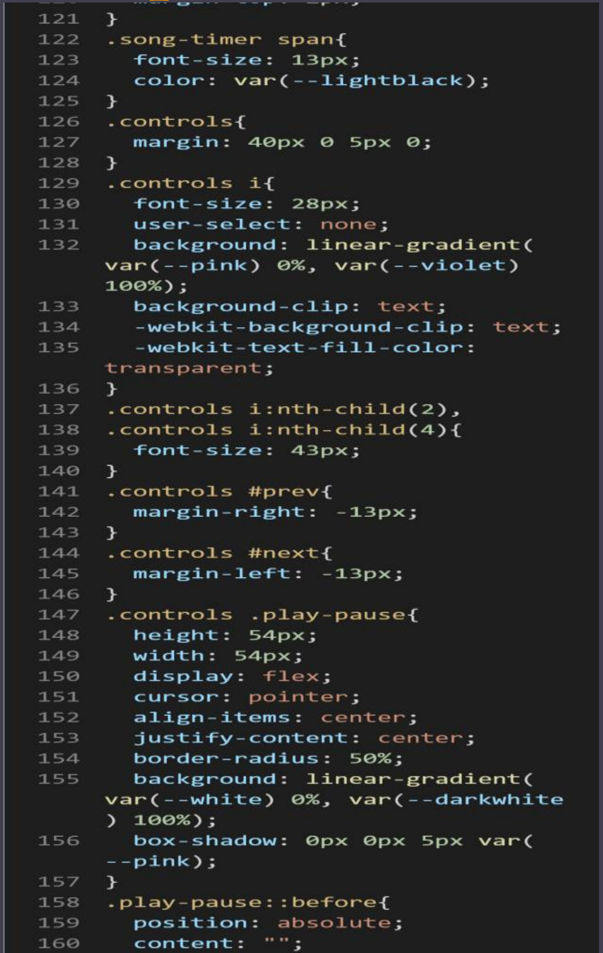


```css
.controls, .music-list .header, .music-list ul li{
  display: flex;
  align-items: center;
  justify-content: space-between;
}
.top-bar i{
  font-size: 30px;
  color: var(--lightblack);
}
.top-bar i:first-child{
  margin-left: -7px;
}
.top-bar span{
  font-size: 18px;
  margin-left: -3px;
  color: var(--lightblack);
}
.img-area{
  width: 100%;
  height: 256px;
  overflow: hidden;
  margin-top: 25px;
  border-radius: 15px;
  box-shadow: 0px 6px 12px var(--lightbshadow);
}
.img-area img{
  width: 100%;
  height: 100%;
  object-fit: cover;
}
.song-details{
  text-align: center;
  margin: 30px 0;
}
.song-details p{
  color: var(--lightblack);
}
.song-details .name{
  font-size: 21px;
}
```

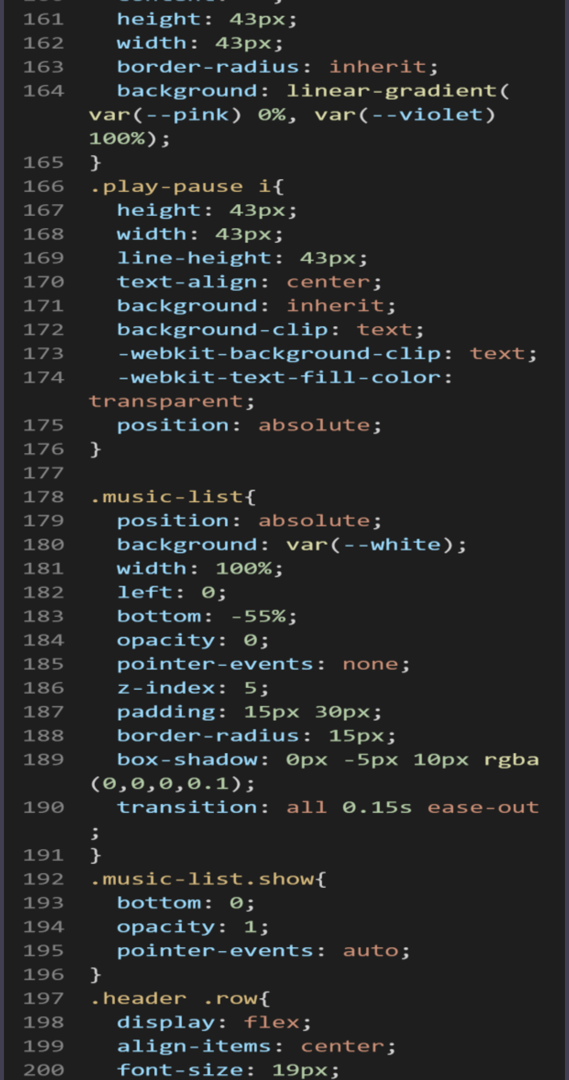


```css
.song-details .artist{
  font-size: 18px;
  opacity: 0.9;
  line-height: 35px;
}
.progress-area{
  height: 6px;
  width: 100%;
  border-radius: 50px;
  background: #f0f0f0;
  cursor: pointer;
}
.progress-area .progress-bar{
  height: inherit;
  width: 0%;
  position: relative;
  border-radius: inherit;
  background: linear-gradient(90deg, var(--pink) 0%, var(--violet) 100%);
}
.progress-bar::before{
  content: "";
  position: absolute;
  height: 12px;
  width: 12px;
  border-radius: 50%;
  top: 50%;
  right: -5px;
  z-index: 2;
  opacity: 0;
  pointer-events: none;
  transform: translateY(-50%);
  background: inherit;
  transition: opacity 0.2s ease;
}
.progress-area:hover .progress-bar::before{
  opacity: 1;
  pointer-events: auto;
}
.progress-area .song-timer{
  margin-top: 2px;
```

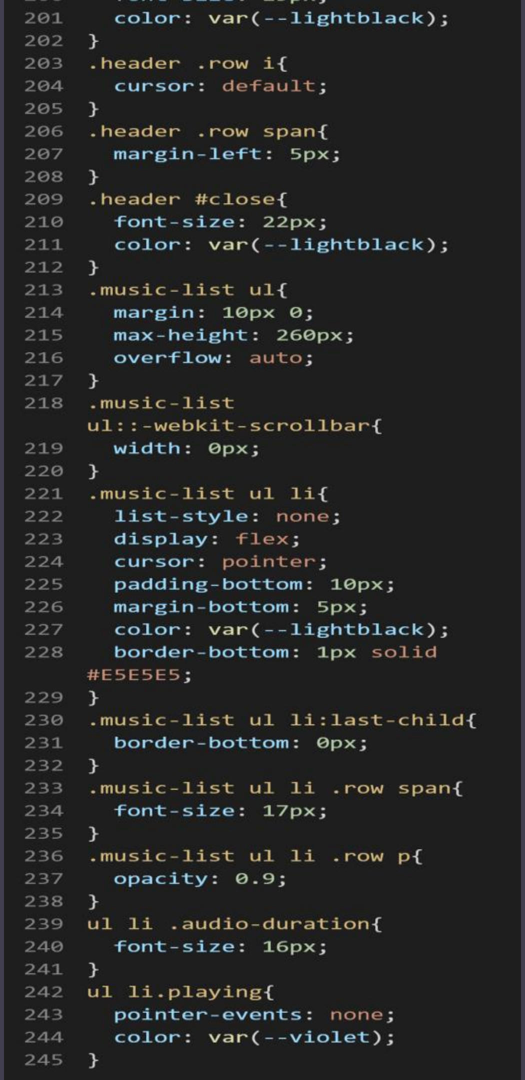
Input:css



```css
}
.song-timer span{
  font-size: 13px;
  color: var(--lightblack);
}
.controls{
  margin: 40px 0 5px 0;
}
.controls i{
  font-size: 28px;
  user-select: none;
  background: linear-gradient(var(--pink) 0%, var(--violet) 100%);
  background-clip: text;
  -webkit-background-clip: text;
  -webkit-text-fill-color: transparent;
}
.controls i:nth-child(2),
.controls i:nth-child(4){
  font-size: 43px;
}
.controls #prev{
  margin-right: -13px;
}
.controls #next{
  margin-left: -13px;
}
.controls .play-pause{
  height: 54px;
  width: 54px;
  display: flex;
  cursor: pointer;
  align-items: center;
  justify-content: center;
  border-radius: 50%;
  background: linear-gradient(var(--white) 0%, var(--darkwhite) 100%);
  box-shadow: 0px 0px 5px var(--pink);
}
.play-pause::before{
  position: absolute;
  content: "";
```

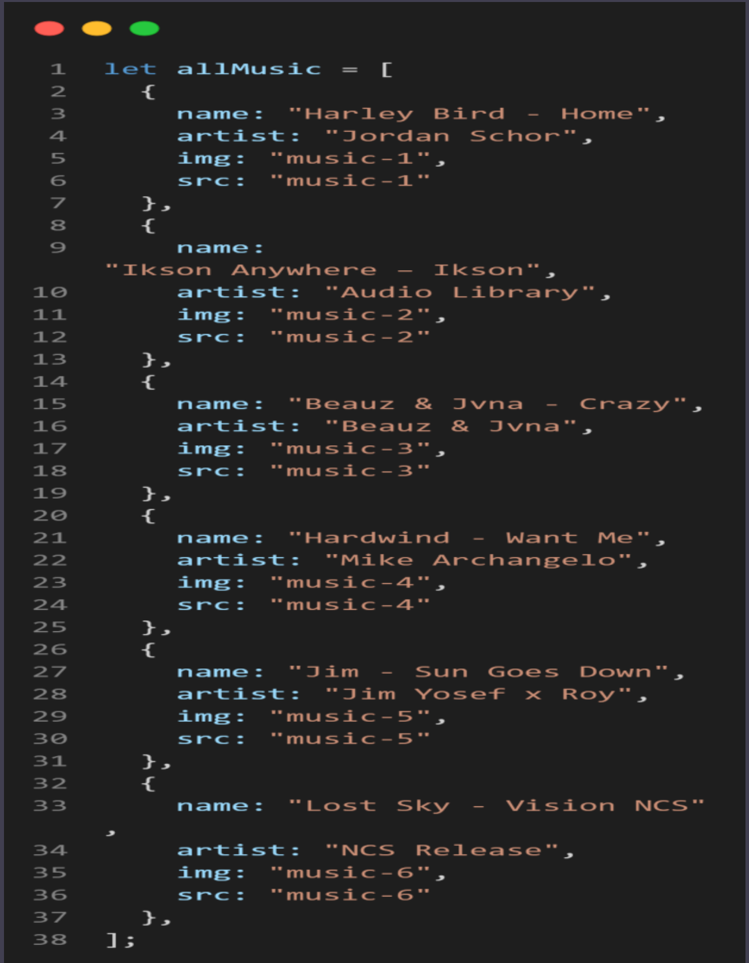


```css
  height: 43px;
  width: 43px;
  border-radius: inherit;
  background: linear-gradient(var(--pink) 0%, var(--violet) 100%);
}
.play-pause i{
  height: 43px;
  width: 43px;
  line-height: 43px;
  text-align: center;
  background: inherit;
  background-clip: text;
  -webkit-background-clip: text;
  -webkit-text-fill-color: transparent;
  position: absolute;
}

.music-list{
  position: absolute;
  background: var(--white);
  width: 100%;
  left: 0;
  bottom: -55%;
  opacity: 0;
  pointer-events: none;
  z-index: 5;
  padding: 15px 30px;
  border-radius: 15px;
  box-shadow: 0px -5px 10px rgba(0,0,0,0.1);
  transition: all 0.15s ease-out;
}
.music-list.show{
  bottom: 0;
  opacity: 1;
  pointer-events: auto;
}
.header .row{
  display: flex;
  align-items: center;
  font-size: 19px;
```



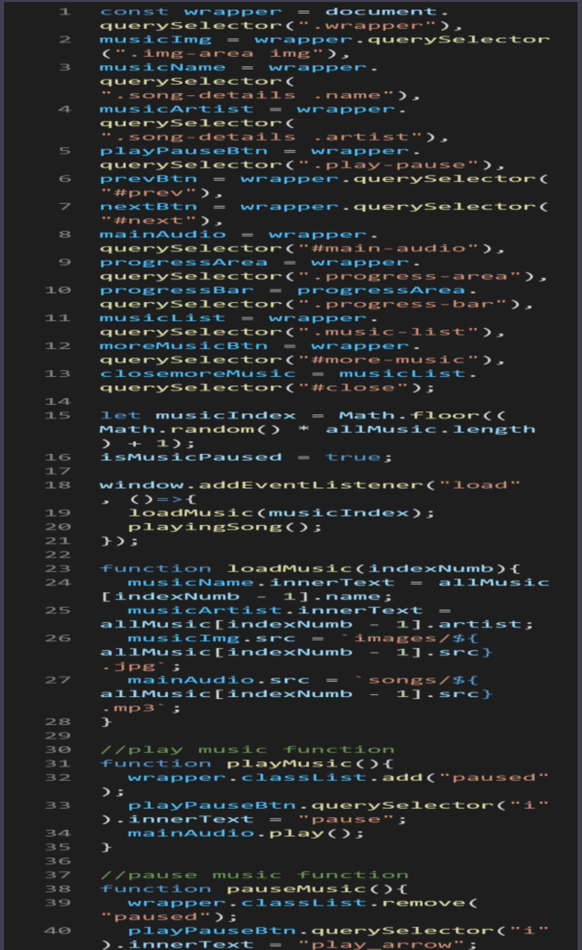
```css
  color: var(--lightblack);
}
.header .row i{
  cursor: default;
}
.header .row span{
  margin-left: 5px;
}
.header #close{
  font-size: 22px;
  color: var(--lightblack);
}
.music-list ul{
  margin: 10px 0;
  max-height: 260px;
  overflow: auto;
}
.music-list ul::-webkit-scrollbar{
  width: 0px;
}
.music-list ul li{
  list-style: none;
  display: flex;
  cursor: pointer;
  padding-bottom: 10px;
  margin-bottom: 5px;
  color: var(--lightblack);
  border-bottom: 1px solid #E5E5E5;
}
.music-list ul li:last-child{
  border-bottom: 0px;
}
.music-list ul li .row span{
  font-size: 17px;
}
.music-list ul li .row p{
  opacity: 0.9;
}
ul li .audio-duration{
  font-size: 16px;
}
ul li.playing{
  pointer-events: none;
  color: var(--violet);
}
```

Input:Music List

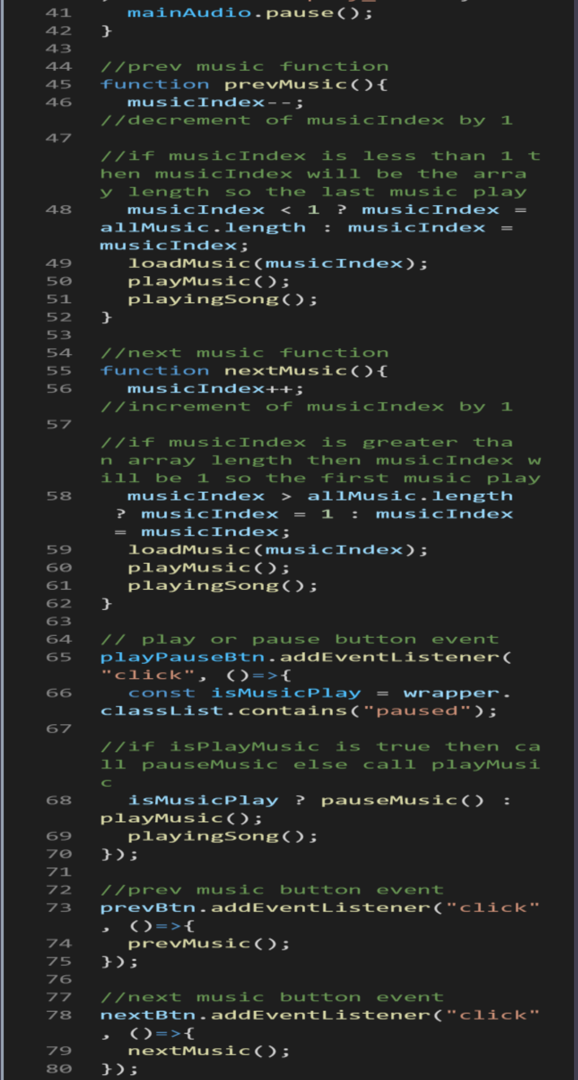


```
let allMusic = [
  {
    name: "Harley Bird - Home",
    artist: "Jordan Schor",
    img: "music-1",
    src: "music-1"
  },
  {
    name: "Ikson Anywhere – Ikson",
    artist: "Audio Library",
    img: "music-2",
    src: "music-2"
  },
  {
    name: "Beauz & Jvna - Crazy",
    artist: "Beauz & Jvna",
    img: "music-3",
    src: "music-3"
  },
  {
    name: "Hardwind - Want Me",
    artist: "Mike Archangelo",
    img: "music-4",
    src: "music-4"
  },
  {
    name: "Jim - Sun Goes Down",
    artist: "Jim Yosef x Roy",
    img: "music-5",
    src: "music-5"
  },
  {
    name: "Lost Sky - Vision NCS",
    artist: "NCS Release",
    img: "music-6",
    src: "music-6"
  },
];
```

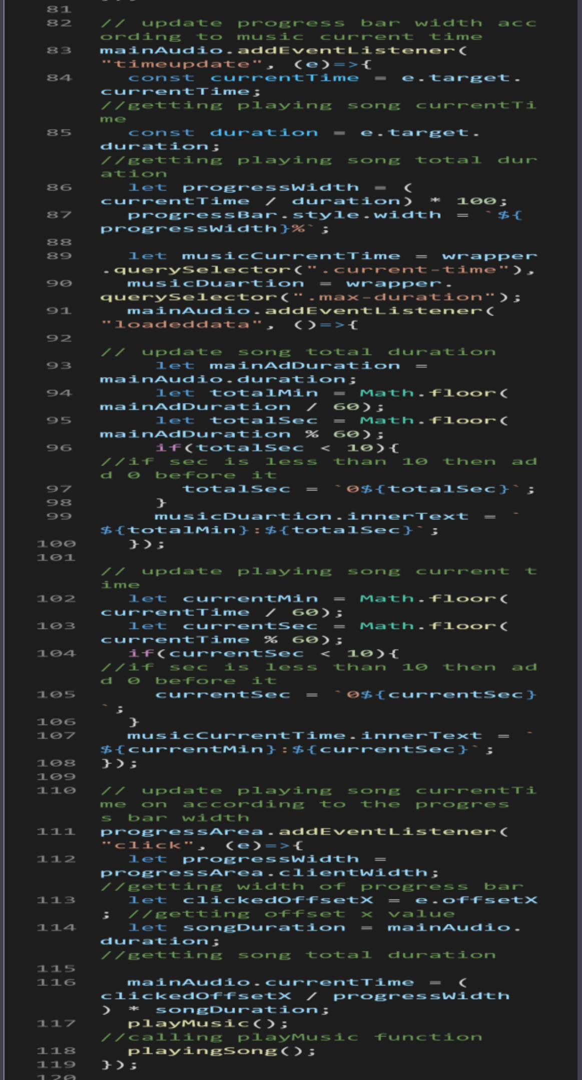
Input:JavaScript



```javascript
const wrapper = document.querySelector(".wrapper"),
musicImg = wrapper.querySelector(".img-area img"),
musicName = wrapper.querySelector(".song-details .name"),
musicArtist = wrapper.querySelector(".song-details .artist"),
playPauseBtn = wrapper.querySelector(".play-pause"),
prevBtn = wrapper.querySelector("#prev"),
nextBtn = wrapper.querySelector("#next"),
mainAudio = wrapper.querySelector("#main-audio"),
progressArea = wrapper.querySelector(".progress-area"),
progressBar = progressArea.querySelector(".progress-bar"),
musicList = wrapper.querySelector(".music-list"),
moreMusicBtn = wrapper.querySelector("#more-music"),
closemoreMusic = musicList.querySelector("#close");

let musicIndex = Math.floor((Math.random() * allMusic.length) + 1);
isMusicPaused = true;

window.addEventListener("load", ()=>{
  loadMusic(musicIndex);
  playingSong();
});

function loadMusic(indexNumb){
  musicName.innerText = allMusic[indexNumb - 1].name;
  musicArtist.innerText = allMusic[indexNumb - 1].artist;
  musicImg.src = `images/${allMusic[indexNumb - 1].src}.jpg`;
  mainAudio.src = `songs/${allMusic[indexNumb - 1].src}.mp3`;
}

//play music function
function playMusic(){
  wrapper.classList.add("paused");
  playPauseBtn.querySelector("i").innerText = "pause";
  mainAudio.play();
}

//pause music function
function pauseMusic(){
  wrapper.classList.remove("paused");
  playPauseBtn.querySelector("i").innerText = "play_arrow";
```

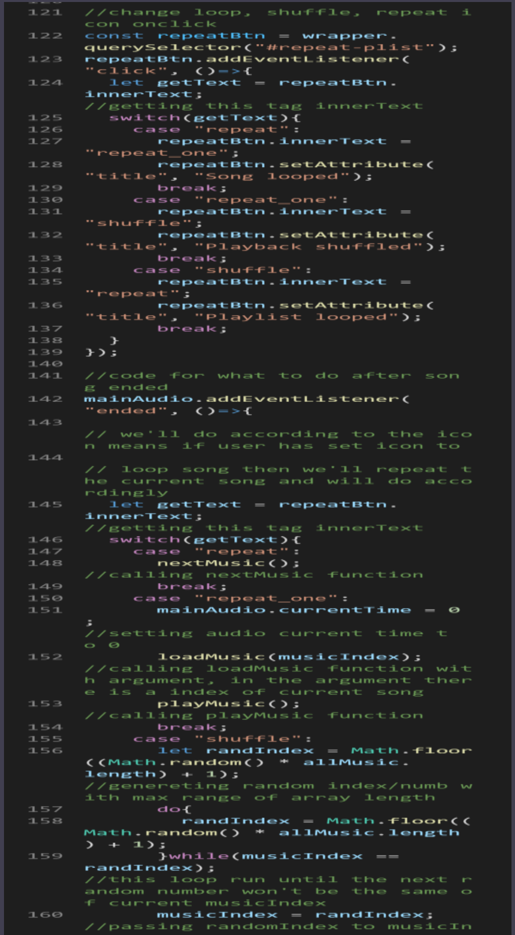


```javascript
  mainAudio.pause();
}

//prev music function
function prevMusic(){
  musicIndex--;
//decrement of musicIndex by 1

//if musicIndex is less than 1 then musicIndex will be the array length so the last music play
  musicIndex < 1 ? musicIndex = allMusic.length : musicIndex = musicIndex;
  loadMusic(musicIndex);
  playMusic();
  playingSong();
}

//next music function
function nextMusic(){
  musicIndex++;
//increment of musicIndex by 1

//if musicIndex is greater than array length then musicIndex will be 1 so the first music play
  musicIndex > allMusic.length ? musicIndex = 1 : musicIndex = musicIndex;
  loadMusic(musicIndex);
  playMusic();
  playingSong();
}

// play or pause button event
playPauseBtn.addEventListener("click", ()=>{
  const isMusicPlay = wrapper.classList.contains("paused");

//if isPlayMusic is true then call pauseMusic else call playMusic
  isMusicPlay ? pauseMusic() : playMusic();
  playingSong();
});

//prev music button event
prevBtn.addEventListener("click", ()=>{
  prevMusic();
});

//next music button event
nextBtn.addEventListener("click", ()=>{
  nextMusic();
});
```

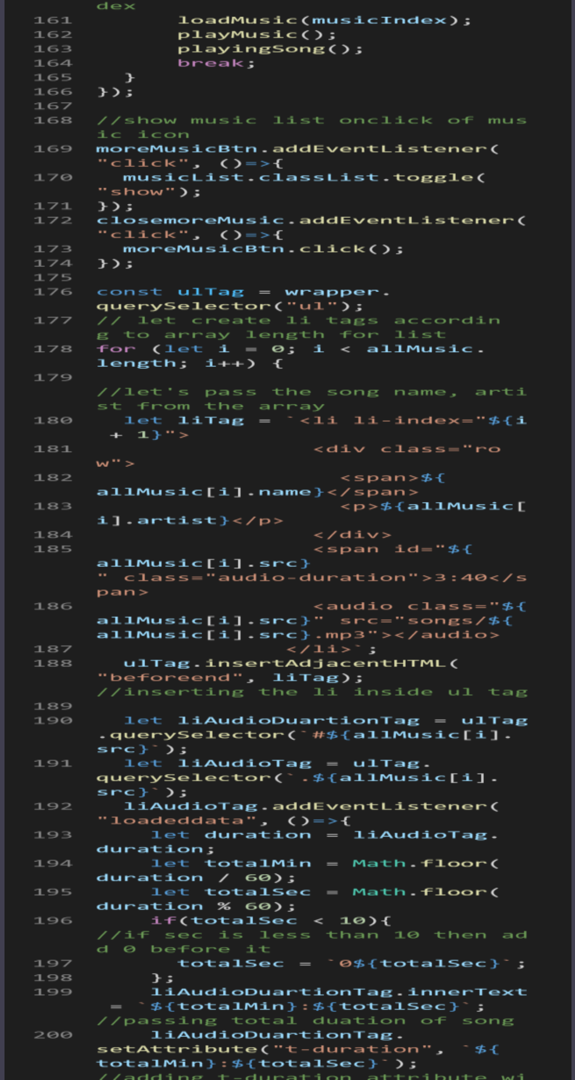


```javascript

// update progress bar width according to music current time
mainAudio.addEventListener("timeupdate", (e)=>{
  const currentTime = e.target.currentTime;
//getting playing song currentTime
  const duration = e.target.duration;
//getting playing song total duration
  let progressWidth = (currentTime / duration) * 100;
  progressBar.style.width = `${progressWidth}%`;

  let musicCurrentTime = wrapper.querySelector(".current-time"),
  musicDuartion = wrapper.querySelector(".max-duration");
  mainAudio.addEventListener("loadeddata", ()=>{

// update song total duration
    let mainAdDuration = mainAudio.duration;
    let totalMin = Math.floor(mainAdDuration / 60);
    let totalSec = Math.floor(mainAdDuration % 60);
    if(totalSec < 10){
//if sec is less than 10 then add 0 before it
      totalSec = `0${totalSec}`;
    }
    musicDuartion.innerText = `${totalMin}:${totalSec}`;
  });

// update playing song current time
  let currentMin = Math.floor(currentTime / 60);
  let currentSec = Math.floor(currentTime % 60);
  if(currentSec < 10){
//if sec is less than 10 then add 0 before it
    currentSec = `0${currentSec}`;
  }
  musicCurrentTime.innerText = `${currentMin}:${currentSec}`;
});

// update playing song currentTime on according to the progress bar width
progressArea.addEventListener("click", (e)=>{
  let progressWidth = progressArea.clientWidth;
//getting width of progress bar
  let clickedOffsetX = e.offsetX; //getting offset x value
  let songDuration = mainAudio.duration;
//getting song total duration

  mainAudio.currentTime = (clickedOffsetX / progressWidth) * songDuration;
  playMusic();
//calling playMusic function
  playingSong();
});
```

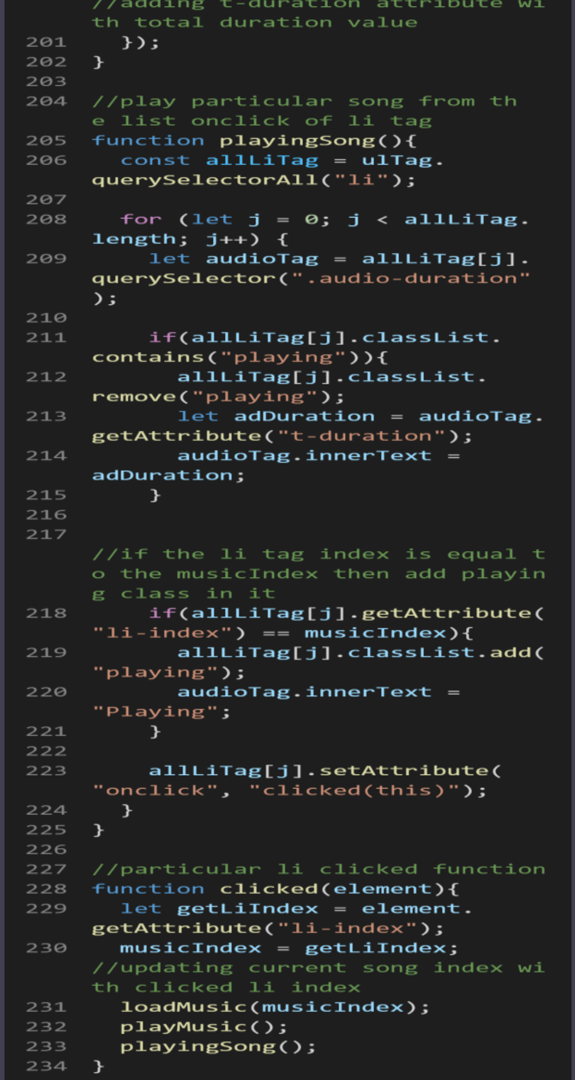
Input:JavaScript



```
//change loop, shuffle, repeat i
con onclick
const repeatBtn = wrapper.
querySelector("#repeat-plist");
repeatBtn.addEventListener(
"click", ()=>{
  let getText = repeatBtn.
innerText;
//getting this tag innerText
  switch(getText){
    case "repeat":
      repeatBtn.innerText =
"repeat_one";
      repeatBtn.setAttribute(
"title", "Song looped");
      break;
    case "repeat_one":
      repeatBtn.innerText =
"shuffle";
      repeatBtn.setAttribute(
"title", "Playback shuffled");
      break;
    case "shuffle":
      repeatBtn.innerText =
"repeat";
      repeatBtn.setAttribute(
"title", "Playlist looped");
      break;
  }
});

//code for what to do after son
g ended
mainAudio.addEventListener(
"ended", ()=>{

// we'll do according to the ico
n means if user has set icon to

// loop song then we'll repeat t
he current song and will do acco
rdingly
  let getText = repeatBtn.
innerText;
//getting this tag innerText
  switch(getText){
    case "repeat":
      nextMusic();
//calling nextMusic function
      break;
    case "repeat_one":
      mainAudio.currentTime = 0
;
//setting audio current time t
o 0
      loadMusic(musicIndex);
//calling loadMusic function wit
h argument, in the argument ther
e is a index of current song
      playMusic();
//calling playMusic function
      break;
    case "shuffle":
      let randIndex = Math.floor
((Math.random() * allMusic.
length) + 1);
//genereting random index/numb w
ith max range of array length
      do{
        randIndex = Math.floor((
Math.random() * allMusic.length
) + 1);
      }while(musicIndex ==
randIndex);
//this loop run until the next r
andom number won't be the same o
f current musicIndex
      musicIndex = randIndex;
//passing randomIndex to musicIn
```

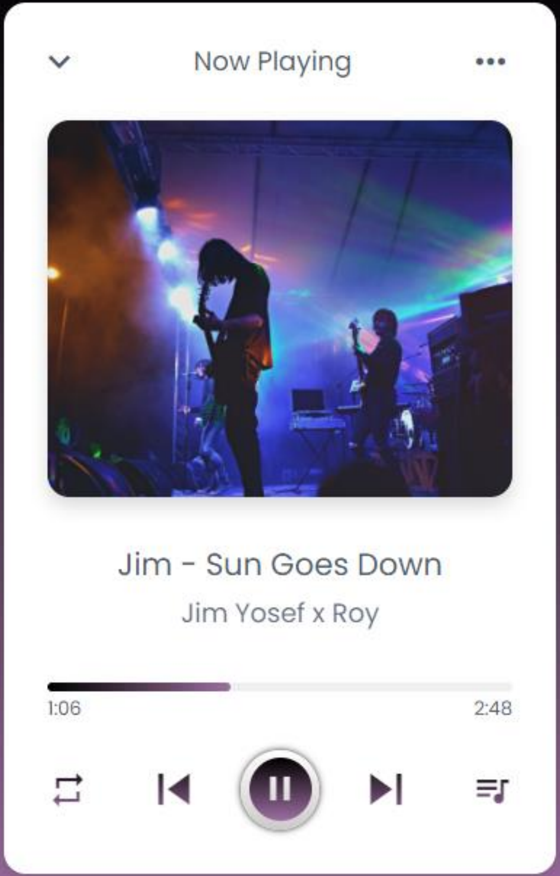


```
dex
      loadMusic(musicIndex);
      playMusic();
      playingSong();
      break;
  }
});

//show music list onclick of mus
ic icon
moreMusicBtn.addEventListener(
"click", ()=>{
  musicList.classList.toggle(
"show");
});
closemoreMusic.addEventListener(
"click", ()=>{
  moreMusicBtn.click();
});

const ulTag = wrapper.
querySelector("ul");
// let create li tags accordin
g to array length for list
for (let i = 0; i < allMusic.
length; i++) {

//let's pass the song name, arti
st from the array
  let liTag = `<li li-index="${i
 + 1}">
                <div class="ro
w">
                  <span>${
allMusic[i].name}</span>
                  <p>${allMusic[
i].artist}</p>
                </div>
                <span id="${
allMusic[i].src}
" class="audio-duration">3:40</s
pan>
                <audio class="${
allMusic[i].src}" src="songs/${
allMusic[i].src}.mp3"></audio>
              </li>`;
  ulTag.insertAdjacentHTML(
"beforeend", liTag);
//inserting the li inside ul tag

  let liAudioDuartionTag = ulTag
.querySelector(`#${allMusic[i].
src}`);
  let liAudioTag = ulTag.
querySelector(`.${allMusic[i].
src}`);
  liAudioTag.addEventListener(
"loadeddata", ()=>{
    let duration = liAudioTag.
duration;
    let totalMin = Math.floor(
duration / 60);
    let totalSec = Math.floor(
duration % 60);
    if(totalSec < 10){
//if sec is less than 10 then ad
d 0 before it
      totalSec = `0${totalSec}`;
    };
    liAudioDuartionTag.innerText
 = `${totalMin}:${totalSec}`;
//passing total duation of song
    liAudioDuartionTag.
setAttribute("t-duration", `${
totalMin}:${totalSec}`);
//adding t-duration attribute wi
```



```
//adding t-duration attribute wi
th total duration value
  });
}

//play particular song from th
e list onclick of li tag
function playingSong(){
  const allLiTag = ulTag.
querySelectorAll("li");

  for (let j = 0; j < allLiTag.
length; j++) {
    let audioTag = allLiTag[j].
querySelector(".audio-duration"
);

    if(allLiTag[j].classList.
contains("playing")){
      allLiTag[j].classList.
remove("playing");
      let adDuration = audioTag.
getAttribute("t-duration");
      audioTag.innerText =
adDuration;
    }


//if the li tag index is equal t
o the musicIndex then add playin
g class in it
    if(allLiTag[j].getAttribute(
"li-index") == musicIndex){
      allLiTag[j].classList.add(
"playing");
      audioTag.innerText =
"Playing";
    }

    allLiTag[j].setAttribute(
"onclick", "clicked(this)");
  }
}

//particular li clicked function
function clicked(element){
  let getLiIndex = element.
getAttribute("li-index");
  musicIndex = getLiIndex;
//updating current song index wi
th clicked li index
  loadMusic(musicIndex);
  playMusic();
  playingSong();
}
```

Output:



Now Playing

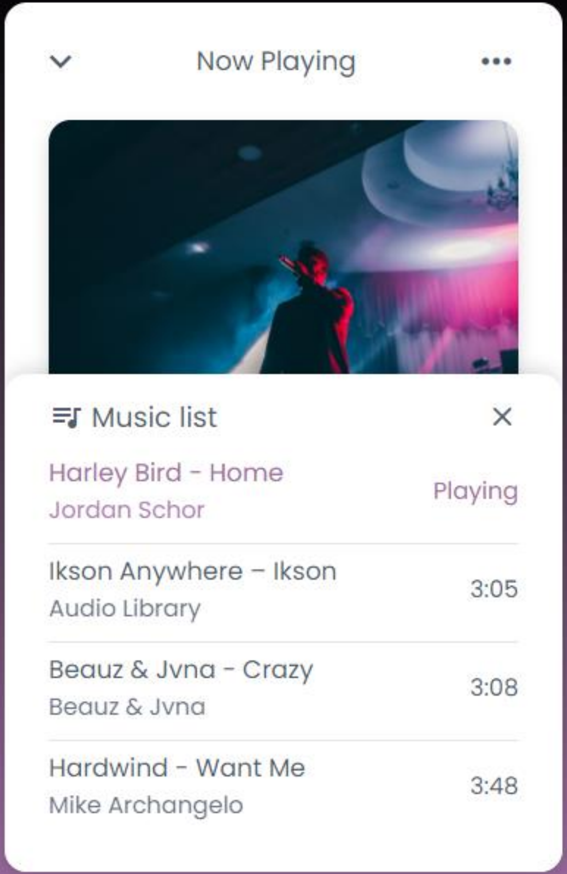
Jim - Sun Goes Down

Jim Yosef x Roy

1:06

2:48

Output:



Music list

Harley Bird - Home
Jordan Schor
Playing

Ikson Anywhere – Ikson
Audio Library
3:05

Beauz & Jvna - Crazy
Beauz & Jvna
3:08

Hardwind - Want Me
Mike Archangelo
3:48

Presented By:

Name-Tejas Saxena
Roll no.-CSE-19/083

Name-Tarun Saxena
Roll no.-CSE-19/082

Thanks