**ABSTRACT**

In order to solve the problem of complex functions and large required memory of mobile phone music player on the current market, a new music player of simple, convenient, less required memory as well as user-friendly is developed. In this paper, we propose a mini music player with a clean user interface purely implemented using HTML, CSS and JavaScript which lead to design and coding of music player. The new design mainly realizes limited core functions including play, pause, volume control with big bold texts highlighted which are used to show the title track playing.It supports audio files and allow users to enjoy music in any browser.

**INDEX**

**S.NO CONTENTS PAGE NO**

**1 INTRODUCTION 4**

1.1 EXISTING SYSTEM 4

1.2 PROPOSED SYSTEM 4

1.3 OVERVIEW 5-8

**2 MODULES 8**

2.1 MODULE DESCRIPTION 8

**3 FUNCTIONALITY 8**

3.1 FEASIBILITY DESIGN 9-11

**4 PHYSICAL DESIGN 12**

4.1 HARDWARE REQUIREMENTS 12

4.2 SOFTWARE REQUIREMENTS 12

4.3 OPERATING SYSTEMS SUPPORTED 12

**5 TECHNOLOGIES USED 13**

5.1 HTML 13

5.2 CSS 13

5.3 JAVASCRIPT 14-15

**6 IMPLEMENTATION 16-28**

**7 TESTING 28-30**

**8 SCREENSHOTS 31-33**

**9 CONCLUSION 34**

**10 BIBLIOGRAPHY 35**

**LIST OF SCREENSHOTS**

**S.NO SCREENSHOT PAGE NO**

**1 MUSICPLAYER 31**

**2 PLAYING SONG 31**

**3 PLAYING NEXT SONG 32**

**4 MOBILE SCREENSHOTS 33**

**1.INTRODUCTION**

* 1. **EXISTING SYSTEM**

As streaming is increasingly being adopted by users, online media players have become essential for consuming media on the internet.

Music players allow one to enjoy music in any browser and supports a lot of the features of an offline music player.

Usually if we want to listen to some music, we need to download an app and then get the songs what we wanted.

It works only in online and if we want to listen in offline we need to download the required songs.

**1.2 PROPOSED SYSTEM**

Creating a music player with a clean user interface that can be used to play music in the browser.

We will also implement features like seeking and volume control.

The songs are directly available in browser and we can listen to them easily. It does not require any app.

**Advantages:**

* It offers low data size.
* **The smaller file size enables the user to rip a large amount of music files.**
* **The compression ratio is not fixed.**

**Disadvantages:**

* There is no proper sorting of songs.
* Searching of a song is not possible here.
  1. **OVERVIEW**

Music player is a system which can run over browsers directly rather than downloading an application.

**HTML**

* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for creating Web pages
* HTML describes the structure of a Web page
* HTML consists of a series of elements
* HTML elements tell the browser how to display the content
* HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.
* The <!DOCTYPE> declaration represents the document type, and helps browsers to display web pages correctly.
* It must only appear once, at the top of the page (before any HTML tags).
* The <!DOCTYPE> declaration is not case sensitive.
* The <div> tag defines a division or a section in an HTML document.
* The <div> tag is used as a container for HTML elements - which is then styled with CSS or manipulated with JavaScript.
* The <div> tag is easily styled by using the class or id attribute.
* Any sort of content can be put inside the <div> tag!

**CSS - CASCADING STYLE SHEETS**

* CSS is the language we use to style an HTML document.
* CSS describes how HTML elements should be displayed.
* 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 web pages all at once
* External stylesheets are stored in **CSS files.**
* CSS SYNTAX: A CSS rule-set consists of a selector and a declaration block:
* 
* The selector points to the HTML element you want to style.
* The declaration block contains one or more declarations separated by semicolons.
* Each declaration includes a CSS property name and a value, separated by a colon.
* Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.
* CSS selectors are used to "find" (or select) the HTML elements you want to style.
* Simple selectors (select elements based on name, id, class)
* [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them)
* [Pseudo-class selectors](https://www.w3schools.com/css/css_pseudo_classes.asp) (select elements based on a certain state)
* [Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element)
* [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)
* There are three ways of inserting a style sheet:
* External CSS
* Internal CSS
* Inline CSS

**JAVASCRIPT**

* JavaScript is the world's most popular programming language.
* JavaScript is the programming language of the Web.
* JavaScript is easy to learn.
* JavaScript is one of the **3 languages** all web developers **must** learn:
* 1. **[HTML](https://www.w3schools.com/html/default.asp)** to define the content of web pages
* 2. **[CSS](https://www.w3schools.com/css/default.asp)** to specify the layout of web pages
* 3. **JavaScript** to program the behavior of web page.
* JavaScript can "display" data in different ways:
* Writing into an HTML element, using innerHTML.
* Writing into the HTML output using document.write().
* Writing into an alert box, using window.alert().
* Writing into the browser console, using console.log().
* To access an HTML element, JavaScript can use the document.getElementById(id) method.
* The id attribute defines the HTML element. The innerHTML property defines the HTML content.
* A **computer program** is a list of "instructions" to be "executed" by a computer.
* In a programming language, these programming instructions are called **statements**.
* A **JavaScript program** is a list of programming **statements**.
* In HTML, JavaScript programs are executed by the web browser.
* JavaScript statements are composed of:
* Values, Operators, Expressions, Keywords, and Comments.
* This statement tells the browser to write "Hello Dolly." inside an HTML element with id="demo":
* JavaScript programs (and JavaScript statements) are often called JavaScript code.
* JavaScript statements can be grouped together in code blocks, inside curly brackets {...}.
* The purpose of code blocks is to define statements to be executed together.
* The JavaScript syntax defines two types of values:
* Fixed values
* Variable values
* Fixed values are called **Literals**
* Variable values are called **Variables**.
* In a programming language, **variables** are used to **store** data values.
* JavaScript uses the var keyword to **declare** variables.
* An **equal sign** is used to **assign values** to variables.

**2. MODULES**

There are four modules:

* Play music
* Pause music
* Change and Alter song
* Volume control

**Modules Description:**

**2.1 Play music**

All the tracks that have to be played are defined in the tracklist as objects. These objects contain properties like the name, artist, image and path to the track. Each of the tracks can then be accessed using its track index

**2.2 Pause music**

A function pauseTrack() handles the playing of the currently loaded track. The pause() method of the HTMLMediaElement API is used for this function. The icon of the button also changes back to the play icon.

**2.3 Change and alter song**

A function prevTrack() handles the loading of the previous track and moving the index backward. The index is reset to the last track when the index reaches the first track. The loadTrack() method defined above is used for loading the new track.

**2.4 Volume controller**

To increase and decrease the volume of the song being played we use volume slider which helps to control the volume

**3. FEASIBILITY STUDY**

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operation Feasibility
* Economical Feasibility

**3.1 TECHNICAL FEASIBILITY**

* The technical issue usually raised during the feasibility stage of the investigation includes the following:
* Does the necessary technology exist to do what is suggested?
* Do the proposed equipment's have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security

**3.2 OPERATIONAL FEASIBILITY**

* Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.
* The operational feasibility assessment focuses on the degree to which the proposed development project fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture and existing business processes.
* To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases.

**3.3 ECONOMICAL FEASIBILITY**

The purpose of an ECONOMIC FEASIBILITY is to demonstrate the net benefit of a proposed project for accepting or disbursing electronic funds/benefits, taking into consideration the benefits and costs to the agency, other state agencies, and the general public as a whole.

**4. PHYSICAL DESIGN**

**4.1 HARDWARE REQUIREMENTS**

* System : PentiumIV 2.4
* Hard Disk : 40 GB.
* RAM : 512 Mb.

**4.2 SOFTWARE REQUIREMENTS**

* HTML & CSS
* JavaScript

**4.3 OPERATING SYSTEMS SUPPORTED**

* Windows 7,10,8
* Android

**5. TECHNOLOGIES USED**

Technologies used in these project are

* HTML
* CSS
* JavaScript

**5.1 HTML**

HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g. HTML) are human readable. Language uses tags to define what manipulation has to be done on the text.

**5.1.1 Features of HTML:**

HTML is the most common used language to write web pages. It has recently gained popularity due to its advantages such as: -

1. It is the language which can be easily understood and can be modified.

2. Effective presentations can be made with the HTML with the help of its all formatting tags.

3. It provides the more flexible way to deign web pages along with the text.

4. Links can also be added to the web pages so it helps the readers to browse the information of their interest.

5. You can display HTML documents on any platforms such as Macintosh, Windows and Linux etc.

6. Graphics, videos and sounds can also be added to the web pages which give an extra attractive look to your web pages.

**5.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language  such as HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

**5.3 JavaScript**

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

JavaScript helps you create really beautiful and crazy fast websites. You can develop your website with a console like look and feel and give your users the best Graphical User Experience.

**5.3.1 Features**

Some of the features are lightweight, dynamic, functional and interpreted. Now we are going to discuss some important features of JavaScript.

* Light Weight Scripting language
* Dynamic Typing
* Object-oriented programming support
* Platform Independent

1. Light Weight Scripting Language

JavaScript is a lightweight scripting language because it is made for data handling at the browser only. Since it is not a general-purpose language so it has a limited set of libraries. Also as it is only meant for client-side execution and that too for web applications, hence the lightweight nature of JavaScript is a great feature.

2. Dynamic Typing

JavaScript supports dynamic typing which means types of the variable are defined based on the stored value. For example, if you declare a variable x then you can store either a string or a Number type value or an array or an object. This is known as dynamic typing

3. Object-Oriented Programming Support

Starting from ES6, the concept of class and OOPs has been more refined. Also, in JavaScript, two important principles with OOP in JavaScript are Object Creation patterns (Encapsulation) and Code Reuse patterns (Inheritance). Although JavaScript developers rarely use this feature but its there for everyone to explore.

4. Platform Independent

This implies that JavaScript is platform-independent or we can say it is portable; which simply means that you can simply write the script once and run it anywhere and anytime. In general, you can write your JavaScript applications and run them on any platform or any browser without affecting the output of the Script.

**6. IMPLEMENTATION**

**6.1** **Index.html**

|  |
| --- |
|  |
|  | <html lang="en"> |
|  | <head> |
|  | <meta charset="UTF-8"> |
|  | <meta name="viewport" content="width=device-width, initial-scale=1.0"> |
|  | <title>Simple Music Player</title> |
|  | <!-- Load FontAwesome icons --> |
|  | <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.13.0/css/all.min.css"> |
|  |  |
|  | <!-- Load the custom CSS style file --> |
|  | <link rel="stylesheet" type="text/css" href="style.css"> |
|  | </head> |
|  | <body> |
|  | <div class="player"> |
|  | <div class="details"> |
|  | <div class="now-playing">PLAYING x OF y</div> |
|  | <div class="track-art"></div> |
|  | <div class="track-name">Track Name</div> |
|  | <div class="track-artist">Track Artist</div> |
|  | </div> |
|  | <div class="buttons"> |
|  | <div class="prev-track" onclick="prevTrack()"><i class="fa fa-step-backward fa-2x"></i></div> |
|  | <div class="playpause-track" onclick="playpauseTrack()"><i class="fa fa-play-circle fa-5x"></i></div> |
|  | <div class="next-track" onclick="nextTrack()"><i class="fa fa-step-forward fa-2x"></i></div> |
|  | </div> |
|  | <div class="slider\_container"> |
|  | <div class="current-time">00:00</div> |
|  | <input type="range" min="1" max="100" value="0" class="seek\_slider" onchange="seekTo()"> |
|  | <div class="total-duration">00:00</div> |
|  | </div> |
|  | <div class="slider\_container"> |
|  | <i class="fa fa-volume-down"></i> |
|  | <input type="range" min="1" max="100" value="99" class="volume\_slider" onchange="setVolume()"> |
|  | <i class="fa fa-volume-up"></i> |
|  | </div> |
|  | </div> |
|  |  |
|  | <!-- Load the main script for the player --> |
|  | <script src="main.js"></script> |
|  | </body> |
|  | </html>  **6.2 Main.js**  let now\_playing = document.querySelector(".now-playing");  let track\_art = document.querySelector(".track-art");  let track\_name = document.querySelector(".track-name");  let track\_artist = document.querySelector(".track-artist");  let playpause\_btn = document.querySelector(".playpause-track");  let next\_btn = document.querySelector(".next-track");  let prev\_btn = document.querySelector(".prev-track");  let seek\_slider = document.querySelector(".seek\_slider");  let volume\_slider = document.querySelector(".volume\_slider");  let curr\_time = document.querySelector(".current-time");  let total\_duration = document.querySelector(".total-duration");  let track\_index = 0;  let isPlaying = false;  let updateTimer;  // Create new audio element  let curr\_track = document.createElement('audio');  // Define the tracks that have to be played  let track\_list = [  {  name: "Night Owl",  artist: "Broke For Free",  image: "https://images.pexels.com/photos/2264753/pexels-photo-2264753.jpeg?auto=compress&cs=tinysrgb&dpr=3&h=250&w=250",  path: "https://files.freemusicarchive.org/storage-freemusicarchive-org/music/WFMU/Broke\_For\_Free/Directionless\_EP/Broke\_For\_Free\_-\_01\_-\_Night\_Owl.mp3"  },  {  name: "Enthusiast",  artist: "Tours",  image: "https://images.pexels.com/photos/3100835/pexels-photo-3100835.jpeg?auto=compress&cs=tinysrgb&dpr=3&h=250&w=250",  path: "https://files.freemusicarchive.org/storage-freemusicarchive-org/music/no\_curator/Tours/Enthusiast/Tours\_-\_01\_-\_Enthusiast.mp3"  },  {  name: "Shipping Lanes",  artist: "Chad Crouch",  image: "https://images.pexels.com/photos/1717969/pexels-photo-1717969.jpeg?auto=compress&cs=tinysrgb&dpr=3&h=250&w=250",  path: "https://files.freemusicarchive.org/storage-freemusicarchive-org/music/ccCommunity/Chad\_Crouch/Arps/Chad\_Crouch\_-\_Shipping\_Lanes.mp3",  },  ];  function random\_bg\_color() {  // Get a number between 64 to 256 (for getting lighter colors)  let red = Math.floor(Math.random() \* 256) + 64;  let green = Math.floor(Math.random() \* 256) + 64;  let blue = Math.floor(Math.random() \* 256) + 64;  // Construct a color withe the given values  let bgColor = "rgb(" + red + "," + green + "," + blue + ")";  // Set the background to that color  document.body.style.background = bgColor;  }  function loadTrack(track\_index) {  clearInterval(updateTimer);  resetValues();  curr\_track.src = track\_list[track\_index].path;  curr\_track.load();  track\_art.style.backgroundImage = "url(" + track\_list[track\_index].image + ")";  track\_name.textContent = track\_list[track\_index].name;  track\_artist.textContent = track\_list[track\_index].artist;  now\_playing.textContent = "PLAYING " + (track\_index + 1) + " OF " + track\_list.length;  updateTimer = setInterval(seekUpdate, 1000);  curr\_track.addEventListener("ended", nextTrack);  random\_bg\_color();  }  function resetValues() {  curr\_time.textContent = "00:00";  total\_duration.textContent = "00:00";  seek\_slider.value = 0;  }  // Load the first track in the tracklist  loadTrack(track\_index);  function playpauseTrack() {  if (!isPlaying) playTrack();  else pauseTrack();  }  function playTrack() {  curr\_track.play();  isPlaying = true;  playpause\_btn.innerHTML = '<i class="fa fa-pause-circle fa-5x"></i>';  }  function pauseTrack() {  curr\_track.pause();  isPlaying = false;  playpause\_btn.innerHTML = '<i class="fa fa-play-circle fa-5x"></i>';;  }  function nextTrack() {  if (track\_index < track\_list.length - 1)  track\_index += 1;  else track\_index = 0;  loadTrack(track\_index);  playTrack();  }  function prevTrack() {  if (track\_index > 0)  track\_index -= 1;  else track\_index = track\_list.length;  loadTrack(track\_index);  playTrack();  }  function seekTo() {  seekto = curr\_track.duration \* (seek\_slider.value / 100);  curr\_track.currentTime = seekto;  }  function setVolume() {  curr\_track.volume = volume\_slider.value / 100;  }  function seekUpdate() {  let seekPosition = 0;  if (!isNaN(curr\_track.duration)) {  seekPosition = curr\_track.currentTime \* (100 / curr\_track.duration);  seek\_slider.value = seekPosition;  let currentMinutes = Math.floor(curr\_track.currentTime / 60);  let currentSeconds = Math.floor(curr\_track.currentTime - currentMinutes \* 60);  let durationMinutes = Math.floor(curr\_track.duration / 60);  let durationSeconds = Math.floor(curr\_track.duration - durationMinutes \* 60);  if (currentSeconds < 10) { currentSeconds = "0" + currentSeconds; }  if (durationSeconds < 10) { durationSeconds = "0" + durationSeconds; }  if (currentMinutes < 10) { currentMinutes = "0" + currentMinutes; }  if (durationMinutes < 10) { durationMinutes = "0" + durationMinutes; }  curr\_time.textContent = currentMinutes + ":" + currentSeconds;  total\_duration.textContent = durationMinutes + ":" + durationSeconds;  }  }  **6.3 Style.Css**  body {  background-color: lightgreen;  }  .player {  height: 95vh;  display: flex;  align-items: center;  flex-direction: column;  justify-content: center;  }  .details {  display: flex;  align-items: center;  flex-direction: column;  justify-content: center;  margin-top: 25px;  }  .track-art {  margin: 25px;  height: 250px;  width: 250px;  background-image: url("https://images.pexels.com/photos/262034/pexels-photo-262034.jpeg?auto=compress&cs=tinysrgb&dpr=3&h=750&w=1260");  background-size: cover;  border-radius: 15%;  }  .now-playing {  font-size: 1rem;  }  .track-name {  font-size: 3rem;  }  .track-artist {  font-size: 1.5rem;  }  .buttons {  display: flex;  flex-direction: row;  align-items: center;  }  .playpause-track, .prev-track, .next-track {  padding: 25px;  opacity: 0.8;  }  .playpause-track:hover, .prev-track:hover, .next-track:hover {  opacity: 1.0;  }  .slider\_container {  width: 75%;  max-width: 400px;  display: flex;  justify-content: center;  align-items: center;  }  /\* Modify the appearance of the slider \*/  .seek\_slider, .volume\_slider {  -webkit-appearance: none;  -moz-appearance: none;  appearance: none;  height: 5px;  background: black;  opacity: 0.7;  -webkit-transition: .2s;  transition: opacity .2s;  }  /\* Modify the appearance of the slider thumb \*/  .seek\_slider::-webkit-slider-thumb, .volume\_slider::-webkit-slider-thumb {  -webkit-appearance: none;  -moz-appearance: none;  appearance: none;  width: 15px;  height: 15px;  background: white;  cursor: pointer;  border-radius: 50%;  }  .seek\_slider:hover, .volume\_slider:hover {  opacity: 1.0;  }  .seek\_slider {  width: 60%;  }  .volume\_slider {  width: 30%;  }  .current-time, .total-duration {  padding: 10px;  }  i.fa-volume-down, i.fa-volume-up {  padding: 10px;  }  i.fa-play-circle, i.fa-pause-circle, i.fa-step-forward, i.fa-step-backward {  cursor: pointer;  } |

**7. TESTING**

**7.1 What is Testing?**

A process of executing a program with the explicit intention of finding errors, that is making the program fail.

**7.2 Software Testing:**

It is the process of testing the functionality and correctness of software by running it. Process of the executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as yet undiscovered error. A successful test is one that uncovers an as yet undiscovered error.

Software Testing is usually performed for one of two reasons:

• Defect detection

• Reliability estimation

**7.3** **TESTING STRATEGIES**

**7.3.1 Unit testing**

 Unit testing involves the design of test cases that validate that the  internal program logic is functioning properly, and that program inputs produce valid  outputs. All decision branches and internal code flow should be validated. It is the  testing of individual software units of the application .it is done after the completion  of an individual unit before integration.

**7.3.2 Integration testing**

 Integration tests are designed to test integrated software  components to determine if they actually run as one program. Testing is event driven  and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown  by successfully unit testing, the combination of components is correct and  consistent. Integration testing is specifically aimed at exposing the problems that  arise from the combination of components.

**7.3.3 Functional test**

 Functional tests provide systematic demonstrations that functions tested  are available as specified by the business and technical requirements, system  documentation, and user manuals.

**7.3.4 System Test**

 System testing ensures that the entire integrated software system meets  requirements. It tests a configuration to ensure known and predictable results. An  example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven  process links and integration points.

**7.3.5 White Box Testing**

 White Box Testing is a testing in which in which the software tester has  knowledge of the inner workings, structure and language of the software, or at least  its purpose. It is purpose. It is used to test areas that cannot be reached from a black  box level.

**7.3.6 Black Box Testing**

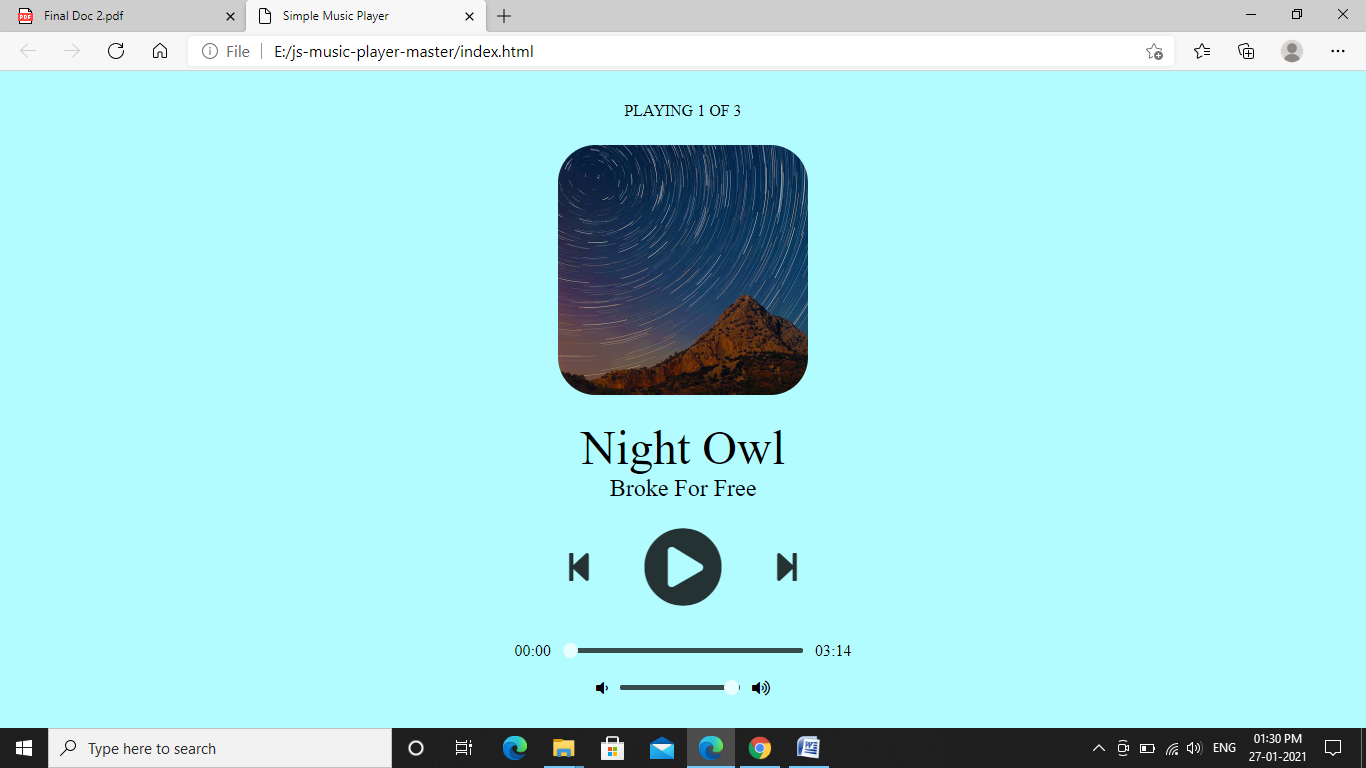
 Black Box Testing is testing the software without any knowledge of  the inner workings, structure or language of the module being tested. Black box tests,  as most other kinds of tests, must be written from a definitive source document, such  as specification or requirements document, such as specification or requirements  document. It is a testing in which the software under test is treated, as a black box.  you cannot “see” into it. The test provides inputs and responds to outputs without  considering how the software works.

**7.4 TEST CASES**

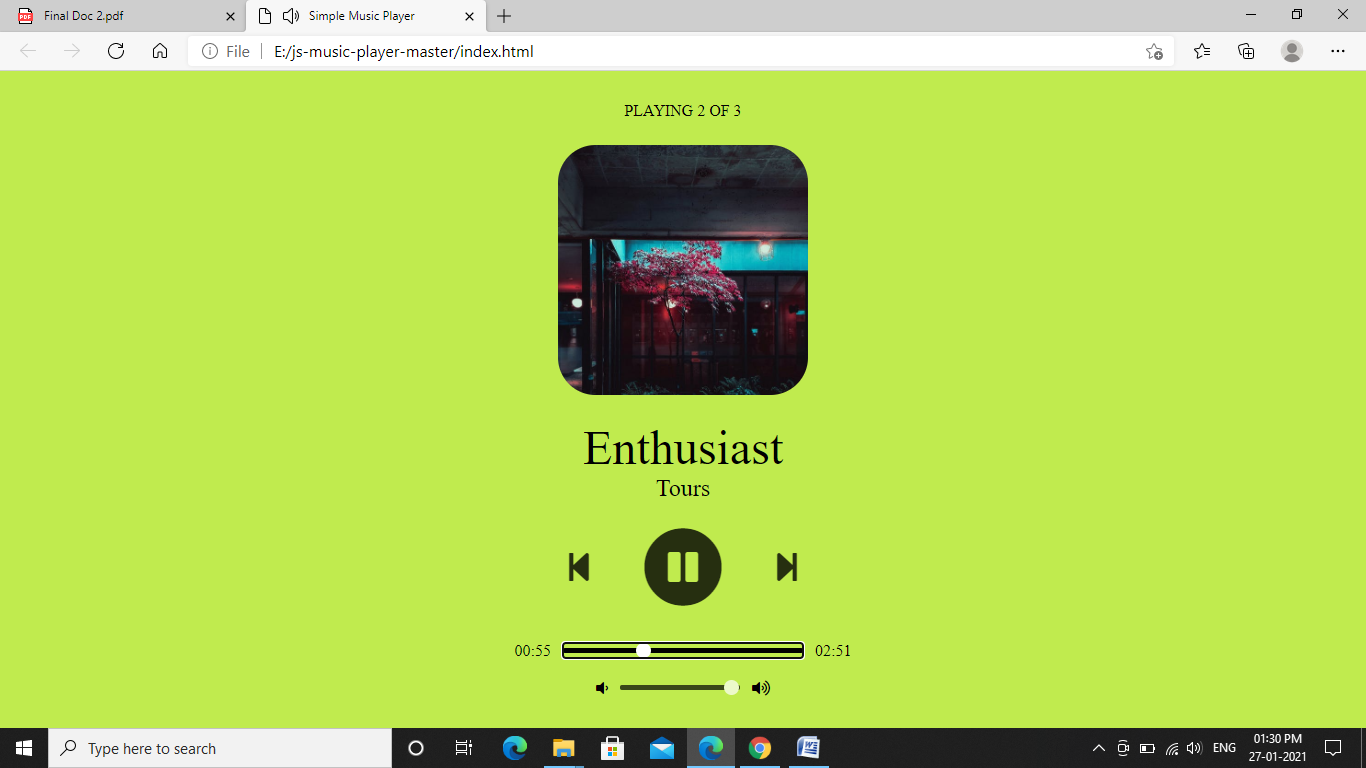
Having test cases that are good at revealing the presence of faults is central to successful testing. The reason for this is that if there is a fault in a program, the program can still provide the expected behavior for many inputs. Only for the set of inputs that exercise the fault in the program will the output of the program deviate from the expected behavior. Hence, it is fair to say that testing is as its test cases

**8. SCREEN SHOTS**

* 1. **Music Player**



* 1. **Playing song**

****

* 1. **Playing Next Song**



**8.4** **Mobile Screen Shots:**



**9.CONCLUSION**

From the above discussion we conclude that the project Music Player System has successfully satisfied the requirements of the users. This project helps the user to play the music without requirement of any application or downloading the songs. This project can run over browser and user can listen to music anywhere and anytime.

**10. BIBLIOGRAPHY**

* Used GOOGLE Search engine for all searches
* t,The Complete ReferenceHerbertSchild Java2,5thEdition, Herbert Schildt(2002), Tata McGrw-HillPublishing Company Ltd
* HTML5 W3C Recommendation 28 October 201436 (@ [http://www.w3.org/TR/html5](http://www.w3.org/TR/html5/)).
* CSS 2.1 Specification W3C Recommendation Revised 17 December 2014 (@ <http://www.w3.org/TR/CSS21/>).
* Matthew MacDonald, "Creating a Website - The Missing Manual", 3rd ed, 2011, O’Reilly.  
  (A good introductory book on HTML/CSS. A new version is expected in July 2015.)
* Matthew MacDonald, "HTML 5 - The Missing Manual", 2nd ed, 2014, O’Reilly.
* David Sawyer McFarland, "CSS 3 - The Missing Manual", 3rd ed, 2013, O’Reilly.
* ECMAScript (JavaScript) Specification: "[Standard ECMA-262 ECMAScript Language Specification 5.1](http://www.ecma-international.org/publications/standards/Ecma-262.htm)", (same as "ISO/IEC 16262" 3rd eds).
* David Sawyer McFarland, "JavaScript and jQuery - The missing manual", 3rd ed, 2014, O'Reilly.
* Jonathan Chaffer and Karl Swedbery, "Learning jQuery", 4th ed, 2013, Packt Publishing.