To Submitted -

A

Project Report On

## “Music Player Using HTML CSS & JavaScript”

Amrutvahini Polytechnic, Sangamner Department: - Information Technology

In Partial Fulfilment of the Requirement for the Diploma in - Information Technology

Subject: Client Side Scripting 22519

Submitted By -

|  |  |
| --- | --- |
| **Roll no** | **Name of Student** |
| 47 | Hande Vedant Machhindra |
| 44 | Gunjal Prathamesh Suresh |
| 35 | Ghodekar Suraj Gorakh |

Under The Guidance of

### Prof. Jondhale D.R



Amrutvahini Polytechnic, Sangamner

(Approved by AICTE, NEW DELHI Affiliated MSBTE) 2023-24

Amrutvahini Polytechnic Sangamner,



### Certificate -

Department: -Information Technology

This is to that the Project Report Entitled,

## “Music Player Using HTML CSS & JavaScript”

Is a Benefited Work Carrier Out By -

|  |  |
| --- | --- |
| **Roll no** | **Name of Student** |
| 47 | Hande Vedant Machhindra |
| 44 | Gunjal Prathamesh Suresh |
| 35 | Ghodekar Suraj Gorakh |

In Partial Fulfillment of the Requirement for the Diploma In

Information Technology During The Academic year 2023-24

*Prof. Jondhale D.R Prof. Chaudhari.N.K. (Project Guide) Hod (IT)*

# ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organization. We would to kind to extend our sincere thanks to all of them.

First and foremost, we want to thanks **Prof. Chaudhari.N.K.** HOD IT) Amrutvahini polytechnic, Sangamner for giving opportunity to work on this project.

We are highly indebted to **Prof. Jondhale D.R** (Project guide) for his guidance and constant supervision as well as foe providing Necessary information regarding the project & also for his support in the Project.

We would like to express our gratitude towards our parents & members of Information Technology department for their kind cooperation and encouragement which help us in completion of this Our thanks and appreciations also go to our colleague in developing.

The project and people who have willingly helped us with their abilities.

|  |  |
| --- | --- |
| **Roll no** | **Name of Student** |
| 47 | Hande Vedant Machhindra |
| 44 | Gunjal Prathamesh Suresh |
| 35 | Ghodekar Suraj Gorakh |

# INDEX

|  |  |
| --- | --- |
| **Sr.no** | **Contents** |
| 1. | Rationale |
| 2. | Aims and Benefits |
| 3. | Course Outcomes |
| 4. | Introduction |
| 5. | Actual Methodology Followed |
| 6. | References |
| 7. | Skills Developed/Learning outcome of this Micro- project |
| 8. | Applications of this Micro-project |

## Micro-Project

## “Music Player Using HTML CSS & JavaScript”

#### Rationale :

There are several reasons why it's important to know how to solve equations. The most important one is the problem solving strategies you learn by working through them. It helps train your brain to think. Solving equations is a way of thinking that you will benefit from unconsciously in other parts of your life.

#### Aims & Benefits:

Aims:

Benefits:

## “Music Player Using HTML CSS & JavaScript”

1. Customizable Design: Tailor the website's look and feel to match your brand or personal style, creating a unique user experience.
2. Interactive Features: Utilize JavaScript to implement interactive elements like music players, playlists, and dynamic content, enhancing user engagement.
3. Responsive Layout: Ensure your site performs well on various devices and screen sizes using CSS for a seamless user experience.
4. Enhanced SEO: Improve search engine visibility with semantic HTML and customizable metadata, attracting more visitors to your site.
5. Performance Optimization: Utilize efficient coding practices and features like lazy loading to ensure fast load times and smooth performance.
6. Course Outcomes:

|  |  |
| --- | --- |
| CI 504.1 | Use3 different program flow control structure for design interactive web pages. |
| CI 504.2 | Exceute3 programs on Arrays and functions in Java script. |
| CI 504.3 | Implement3 event based web forms and handling cookies using Java script. |
| CI 504.4 | Apply3 regular expressions for validations to design interactive webpages. |
| CI 504.5 | Implement3 Menus and navigations in web Pages. |

1. Literature Review:

Java Swing has long been favored by developers for its platform independence, extensive widget library, and event-driven programming model. Its versatility makes it suitable for diverse applications, including management systems like the one demonstrated in this project. The project's architecture, with distinct classes for Medicine, MedicalStore, and the GUI interface, adheres to object-oriented design principles, promoting code modularity and reusability.

Literature on Java GUI development emphasizes the significance of user experience (UX) design, responsive interfaces, and efficient data handling. Effective UX design entails intuitive layouts, error handling mechanisms, and clear navigation paths to enhance user satisfaction and productivity. Techniques such as

layout managers, event listeners, and input validation are pivotal for creating responsive and user-friendly interfaces.

In the domain of management systems, previous studies have explored various functionalities crucial for effective inventory and sales management. These include inventory tracking, pricing calculations, order processing, and reporting features. The medical store management system presented in this project encapsulates many of these functionalities, providing a solid foundation for a practical and usable solution.

While Java Swing remains a robust choice for GUI development, advancements like JavaFX and web-based technologies offer alternative avenues for modern UI design and deployment. Research in software engineering continues to evolve, emphasizing agile methodologies, testing strategies, and user-centered design practices to meet evolving user needs and technological landscapes.

Future enhancements to the medical store management system could explore aspects such as database integration for persistent data storage, user authentication and access control for security, and analytics modules for business insights. Additionally, considering the migration path to newer GUI frameworks or technologies can ensure the system's relevance and longevity in a dynamic software ecosystem.

In summary, the literature underscores the importance of Java GUI development, software engineering best practices, and domain-specific considerations in crafting effective management systems. The medical store management system project serves as a practical example of these principles, highlighting opportunities for further research and innovation in the field.

1. Actual Methodology Followed:

A medical store management system is a software application that helps pharmacies and medical stores manage their inventory, sales, purchases, and billing processes efficiently. It includes features like inventory management, medicine cataloging, sales and billing, purchase management, reporting and analytics, user management with security features, alerts and notifications, and supports integration with other systems for scalability and enhanced functionality.

The methodology followed for developing the medical store management system project involves several key steps typical in software development. Here is an overview of the actual methodology followed-

* 1. Requirement Analysis:
     + Identified the requirements for the medical store management system, including functionalities such as adding medicines, displaying medicines, and calculating bills.
     + Considered user inputs and interactions to design an intuitive user interface.
  2. Design Phase:
     + Designed the class structure based on object-oriented principles, with separate classes for Medicine, Medical Store, and the GUI interface.
     + Defined methods for adding medicines, displaying medicines, and calculating bills within the appropriate classes.
     + Designed the GUI layout using Java Swing components such as JTextArea, JComboBox, JTextField, and JButton
  3. Implementation:
     + Implemented the core functionalities of the system within the respective classes.
     + Utilized Java Swing for creating the graphical user interface, including event listeners for button actions.
     + Implemented error handling and input validation to ensure data integrity and prevent runtime errors.
     + Added initial data for medicines in the MedicalStore constructor.
  4. Testing:
     + Conducted testing to ensure that all functionalities work as expected.
     + Tested edge cases such as entering invalid input for quantity or medicine details.
     + Checked the responsiveness and usability of the GUI components.
     + Verified the accuracy of bill calculations and displayed results.
  5. Refinement and Enhancement:
     + Made refinements based on testing feedback and user experience considerations.
     + Enhanced the GUI by adding a total bill label and a button to dynamically add new medicines.
     + Updated the literature review and documentation to reflect the actual implementation details.
  6. Deployment and Maintenance:
     + Packaged the application into an executable JAR file for distribution.
     + Ensured that the application runs smoothly on different Java Runtime Environments (JREs) without dependencies issues.
     + Documented installation instructions and usage guidelines for end-users.
     + Planned for ongoing maintenance, including updates to accommodate future requirements or technology changes.

This methodology follows a structured approach from requirements gathering to deployment, with an emphasis on iterative development, testing, and refinement. It aligns with industry-standard practices in software development, ensuring a reliable and user-friendly medical store management system.

Packets Used:-

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.text.DecimalFormat; import java.util.ArrayList;

In the medical store management system project, the following main points are utilized:

1. Medicine Class:Represents a medicine item with attributes such as name, price, tax percentage, and discount percentage. Includes a method to calculate the total price.
2. MedicalStore Class: Manages a collection of medicines and provides functionalities like adding medicines, displaying available medicines, and calculating bills.
3. Java Swing GUI: Utilizes Java Swing components to create a graphical user interface for interacting with the system. Components such as JTextArea, JComboBox, JTextField, and JButton are used for displaying data and capturing user input.
4. Functionalities:
   * Display Medicines: Lists all available medicines in the medical store.
   * Add Medicine: Allows users to input details of a new medicine and adds it to the store.
   * Calculate Bill: Calculates the total bill for a selected medicine based on quantity, including tax and discounts.
5. User Interaction:
   * User inputs are captured through text fields and combo boxes in the GUI.
   * Event listeners are used to trigger actions such as displaying medicines, adding new medicines, and calculating bills based on user interactions with buttons.
6. Error Handling:
   * Input validation is implemented to handle invalid user inputs, ensuring data integrity and preventing runtime errors.
   * Error messages are displayed to alert users about incorrect inputs or invalid actions.
     + **Project code explain:**

Let's break down the provided code and explain each part:

* + - 1. Imports:
         * `javax.swing.\*`: Imports all classes from the Swing package, which is used for creating GUI components.
         * `java.awt.\*`: Imports the Abstract Window Toolkit (AWT) classes, which are used for basic GUI functionalities.
         * `java.awt.event.ActionEvent` and `java.awt.event.ActionListener`: Used for handling actions/events in the GUI components.
         * `java.text.DecimalFormat`: Used for formatting decimal numbers.
         * `java.util.ArrayList`: Used for creating and managing ArrayLists.
      2. Medicine Class:
         * Represents a medicine item with attributes such as name, price, tax percentage, and discount percentage.
         * Includes a constructor to initialize the medicine attributes.
         * Provides a method `calculateTotalPrice(int quantity)` to calculate the total price of a medicine based on quantity, tax, and discount.
      3. MedicalStore Class:
         * Manages a collection of Medicine objects using an ArrayList.
         * Includes methods to add medicines (`addMedicine`) and retrieve the list of medicines (`getMedicines`).
      4. MedicalStoreGUI Class (Main GUI Class):
         * Extends `JFrame` to create the main GUI window for the medical store management system.
         * Defines instance variables for GUI components such as JTextArea, JComboBox, JTextField, JLabel, and JButton.
         * Initializes the GUI components, sets their properties, and adds them to the GUI layout using panels and layouts.
         * Implements event handling for buttons such as "Display Medicines," "Calculate Bill," and "Add Medicine" using ActionListener interfaces.
         * Includes methods to perform actions like displaying medicines (`displayMedicines`), calculating bills (`calculateBill`), and adding new medicines (`addNewMedicine`).
         * Uses SwingUtilities.invokeLater to ensure that GUI components are created and updated on the Event Dispatch Thread (EDT), which is necessary for Swing applications.
      5. Main Method:
         * The `main` method starts the application by creating an instance of the `MedicalStoreGUI` class, which initializes and displays the main GUI window.

Overall, the code demonstrates a simple medical store management system GUI using Java Swing, with functionalities for displaying medicines, calculating bills, and adding new medicines dynamically through user interactions. It follows a modular

* + - 1. Code :

HTML -

<!DOCTYPE html>

<html>

  <head>

    <title>VPS Music</title>

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link

      rel="stylesheet"

      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" />

    <link rel="stylesheet" type="text/css" href="style.css" />

  </head>

  <body>

    <div class="main">

      <p id="logo"><i class="fa fa-music"></i>VPS Music</p>

      <!-- show\_song\_number -->

      <div class="show\_song\_no">

        <p id="present">1</p>

        <p>/</p>

        <p id="total"></p>

      </div>

      <!--- left part --->

      <div class="left">

        <!--- song img --->

        <img id="track\_image" />

        <div class="volume">

          <p id="volume\_show">90</p>

          <i

            class="fa fa-volume-up"

            aria-hidden="true"

            onclick="mute\_sound()"

            id="volume\_icon"></i>

          <input

            type="range"

            min="0"

            max="100"

            value="90"

            onchange="volume\_change()"

            id="volume" />

        </div>

      </div>

      <!--- right part --->

      <div class="right">

        <!--- song title & artist name --->

        <div class="song\_detail">

          <!-- <p id="title"></p> -->

          <p id="artist"></p>

        </div>

        <!--- middle part --->

        <div class="middle">

          <button onclick="previous\_song()" id="pre">

            <i class="fa fa-step-backward" aria-hidden="true"></i>

          </button>

          <button onclick="justplay()" id="play">

            <i class="fa fa-play" aria-hidden="true"></i>

          </button>

          <button onclick="next\_song()" id="next">

            <i class="fa fa-step-forward" aria-hidden="true"></i>

          </button>

        </div>

        <!--- song duration part --->

        <div class="duration">

          <input

            type="range"

            min="0"

            max="100"

            value="0"

            id="duration\_slider"

            onchange="change\_duration()" />

          <button id="auto" onclick="autoplay\_switch()">

            Auto Play &nbsp;<i

              class="fa fa-circle-o-notch"

              aria-hidden="true"></i>

          </button>

        </div>

      </div>

    </div>

    <!--- Add javascript file --->

    <script src="script.js"></script></body></html>

CSS

\* {

  margin: 0;

  padding: 0;

  font-family: cursive;

}

body {

  min-height: 100vh;

  display: grid;

  place-items: center;

  background: rgba(0, 0, 0, 0.5);

}

.main {

  /\* margin-top: 15px; \*/

  position: relative;

  height: 80vh;

  width: 80%;

  display: flex;

  align-items: center;

  justify-content: center;

  background: #232427;

  border-radius: 8px;

  box-shadow: -3px -2px 151px 12px rgba(255, 1, 153, 1);

}

.main button {

  padding: 10px 12px;

  margin: 0 10px;

}

.main #logo {

  position: absolute;

  top: 10px;

  left: 30px;

  font-size: 25px;

  color: #ccc;

}

.main #logo i {

  margin-right: 15px;

}

/\* left & right part \*/

.left {

  width: 50%;

  display: flex;

  align-items: center;

  justify-content: center;

  flex-direction: column;

}

/\* song image \*/

.left img {

  height: 300px;

  width: 80%;

  border-radius: 15px;

  object-fit: cover;

  box-shadow: inset 2px 2px 5px rgba(0, 0, 0, 0.5),

    inset -2px -2px 5px rgb(80, 195, 224), 5px 5px 15px rgba(0, 0, 0, 0.3),

    -5px -5px 15px rgba(255, 1, 153, 0.7);

  padding: 5px;

  /\* -3px -2px 151px 12px rgba(255, 1, 153, 0.6) \*/

}

/\* both range slider part \*/

input[type="range"] {

  -webkit-appearance: none;

  width: 50%;

  outline: none;

  height: 10px;

  margin: 0 15px;

  overflow: hidden;

  border-radius: 25px;

}

input[type="range"]::-webkit-slider-thumb {

  -webkit-appearance: none;

  height: 10px;

  width: 10px;

  background: #ff02e1;

  cursor: pointer;

  box-shadow: -415px 0 0 400px #ff02e1;

}

.right input[type="range"] {

  width: 40%;

}

/\* volume part \*/

.left .volume {

  margin-top: 25px;

  width: 80%;

  height: 30px;

  display: flex;

  align-items: center;

  justify-content: center;

  color: #ff02e1;

  /\*  border: 1px solid #fff;\*/

}

.volume input[type="range"] {

  flex: 1;

}

.left .volume p {

  font-weight: bold;

  font-size: 15px;

}

.left .volume i {

  cursor: pointer;

  padding: 8px 12px;

  background: rgba(245, 245, 245, 0.1);

  border-radius: 10px;

}

.left .volume i:hover {

  background: rgba(246, 0, 209, 0.1);

}

.volume #volume\_show {

  padding: 8px 12px;

  margin: 0 5px 0 0;

  background: rgba(245, 245, 245, 0.1);

  border-radius: 10px;

}

/\* right part \*/

.right {

  width: 50%;

  padding: 10px 0;

  display: flex;

  align-items: center;

  flex-direction: column;

}

.right .middle {

  width: 100%;

  display: flex;

  align-items: center;

  justify-content: center;

}

.right .middle button {

  border: none;

  height: 70px;

  width: 70px;

  border-radius: 50%;

  display: flex;

  align-items: center;

  justify-content: center;

  cursor: pointer;

  outline: none;

  transition: 0.5s;

  background: #232427;

  box-shadow: inset 2px 2px 5px rgba(0, 0, 255, 0.5),

    inset -2px -2px 5px rgb(0, 204, 255), 5px 5px 15px rgba(0, 0, 0, 0.3),

    -5px -5px 15px rgba(255, 255, 255, 0.3);

}

.song\_detail {

  position: relative;

  width: 80%;

  overflow: hidden;

  margin-bottom: 6.5em;

  /\*  border: 1px solid #fff;\*/

}

/\* .song\_detail #title {

  text-transform: capitalize;

  color: #fff;

  font-size: 35px; \*/

/\* } \*/

.song\_detail #artist {

  text-transform: capitalize;

  color: #fff;

  font-size: 35px;

  margin-top: 5px;

  justify-content: center;

  /\* align-items: center; \*/

}

.right .duration {

  margin-top: 3em;

  position: relative;

  display: flex;

  align-items: center;

  justify-content: center;

  width: 80%;

  /\* border: 1px solid rgba(2, 240, 200, 0.724); \*/

}

.duration input[type="range"] {

  flex: 10;

  /\* box-shadow: -3px -2px 151px 12px rgba(2, 57, 237, 0.673); \*/

}

.right #auto {

  font-size: 17px;

  text-align: center;

  cursor: pointer;

  border: none;

  padding: 5px 7px;

  color: #04e0fd;

  background: rgba(255, 255, 255, 0.2);

  outline: none;

  border-radius: 100px;

  box-shadow: inset 2px 2px 5px rgba(0, 0, 255, 0.5),

    inset -2px -2px 5px rgb(0, 204, 255), 5px 5px 15px rgba(0, 0, 0, 0.3),

    -5px -5px 15px rgba(255, 255, 255, 0.1);

}

/\* #play {

  background: #148f77;

} \*/

.right button:hover {

  background: #148f77;

}

.right i:before {

  color: #04e0fd;

  font-size: 23px;

}

.show\_song\_no {

  position: absolute;

  top: 10px;

  right: 10px;

  width: 30px;

  height: 20px;

  display: flex;

  align-items: center;

  justify-content: center;

  padding: 8px 12px;

  color: #ebebeb;

  border-radius: 5px;

  background: rgba(255, 255, 255, 0.2);

  box-shadow: inset 2px 2px 5px rgba(0, 0, 0, 0.2),

    inset -2px -2px 5px rgba(255, 255, 255, 0.1),

    5px 5px 15px rgba(0, 0, 0, 0.3), -5px -5px 15px rgba(255, 255, 255, 0.1);

}

.show\_song\_no p:nth-child(2) {

  margin: 0 5px;

}

/\*responsive\*/

@media (max-width: 700px) {

  .main {

    min-height: 100vh;

    width: 100%;

    flex-direction: column;

  }

  .right {

    margin-top: 50px;

    width: 60%;

  }

  .right .duration {

    width: 90%;

  }

  .left {

    margin-top: 5em;

    width: 60%;

  }

  .left img {

    min-width: 90%;

    height: 180px;

  }

  .main #logo {

    display: none;

  }

  .song\_detail {

    position: absolute;

    top: 5px;

    left: 10px;

    width: 80%;

    height: 70px;

  }

  .song\_detail #title {

    font-size: 1.8em;

  }

}

@media (max-width: 500px) {

  .main {

    min-height: 100vh;

    width: 100%;

    flex-direction: column;

  }

  .right {

    margin-top: 50px;

    width: 80%;

  }

  .left {

    margin-top: 5em;

    width: 80%;

}

  .left img {

    min-width: 90%;

    height: 180px;

  }

  .main #logo {

    display: none;

  }

  .song\_detail {

    position: absolute;

    top: 5px;

    left: 10px;

    width: 80%;

    height: 70px;

  }

  .song\_detail #title {

    font-size: 1.5em;

  }

  .song\_detail #artist {

    font-size: 0.8em;

  }

  .right .middle button {

    height: 62px;

    width: 62px;

  }

JAVASCRIPT -

let previous = document.querySelector("#pre");

let play = document.querySelector("#play");

let next = document.querySelector("#next");

// let title = document.getElementById("#title");

let recent\_volume = document.querySelector("#volume");

let volume\_show = document.querySelector("#volume\_show");

let slider = document.querySelector("#duration\_slider");

let show\_duration = document.querySelector("#show\_duration");

let track\_image = document.querySelector("#track\_image");

let auto\_play = document.querySelector("#auto");

let present = document.querySelector("#present");

let total = document.querySelector("#total");

let artist = document.querySelector("#artist");

let timer;

let autoplay = 0;

let index\_no = 0;

let Playing\_song = false;

//create a audio Element

let track = document.createElement("audio");

//All songs list

const All\_song = [

  {

    path: "song1.mp3",

    img: "img1.jpg",

    singer: "AAJ KI RAT - Mr . And Mrs. Mahi",

  },

  {

    path: "song2.mp3",

    img: "img2.jpg",

    singer: " AAYI NAI - Asees Kaur",

  },

  {

    path: "song3.mp3",

    img: "img3.jpg",

    singer: " AGAR HO TUM -  Chandu Champion",

  },

  {

    path: "song4.mp3",

    img: "img4.jpg",

    singer: " HALKI HALKI SI - Sarfira ",

  },

  {

    path: "song5.mp3",

    img: "img5.jpg",

    singer: "  KHUNDAY - Stree 2",

  },

  {

    path: "song6.mp3",

    img: "img6.jpg",

    singer: " MERE MEHBOOB MERE SANAM - Bad News",

  },

  {

    path: "song7.mp3",

    img: "img7.jpg",

    singer: " TU HI HYE CHAMPION - Bad News",

  },

];

// All functions

// function load the track

function load\_track(index\_no) {

  track.src = All\_song[index\_no].path;

  track\_image.src = All\_song[index\_no].img;

  artist.innerHTML = All\_song[index\_no].singer;

  // title.innerText = All\_song[index\_no].name;

  track.load();

  total.innerHTML = All\_song.length;

  present.innerHTML = index\_no + 1;

  timer = setInterval(range\_slider, 1000);

}

load\_track(index\_no);

//mute sound function

function mute\_sound() {

  track.volume = 0;

  volume.value = 0;

  volume\_show.innerHTML = 0;

}

// checking.. the song is playing or not

function justplay() {

  if (Playing\_song == false) {

    playsong();

  } else {

    pausesong();

  }

}

// reset song slider

function reset\_slider() {

  slider.value = 0;

}

// play song

function playsong() {

  track.play();

  Playing\_song = true;

  play.innerHTML = '<i class="fa fa-pause"></i>';

}

//pause song

function pausesong() {

  track.pause();

  Playing\_song = false;

  play.innerHTML = '<i class="fa fa-play"></i>';

}

// next song

function next\_song() {

  if (index\_no < All\_song.length - 1) {

    index\_no += 1;

    load\_track(index\_no);

    playsong();

  } else {

    index\_no = 0;

    load\_track(index\_no);

    playsong();

  }

}

// previous song

function previous\_song() {

  if (index\_no > 0) {

    index\_no -= 1;

    load\_track(index\_no);

    playsong();

  } else {

    index\_no = All\_song.length;

    load\_track(index\_no);

    playsong();

  }

}

// change volume

function volume\_change() {

  volume\_show.innerHTML = recent\_volume.value;

  track.volume = recent\_volume.value / 100;

}

// change slider position

function change\_duration() {

  slider\_position = track.duration \* (slider.value / 100);

  track.currentTime = slider\_position;

}

// autoplay function

function autoplay\_switch() {

  if (autoplay == 1) {

    autoplay = 0;

    auto\_play.style.background = "rgba(255,255,255,0.2)";

  } else {

    autoplay = 1;

    auto\_play.style.background = "#148F77";

  }

}

function range\_slider() {

  let position = 0;

  // update slider position

  if (!isNaN(track.duration)) {

    position = track.currentTime \* (100 / track.duration);

    slider.value = position;

  }

  // function will run when the song is over

  if (track.ended) {

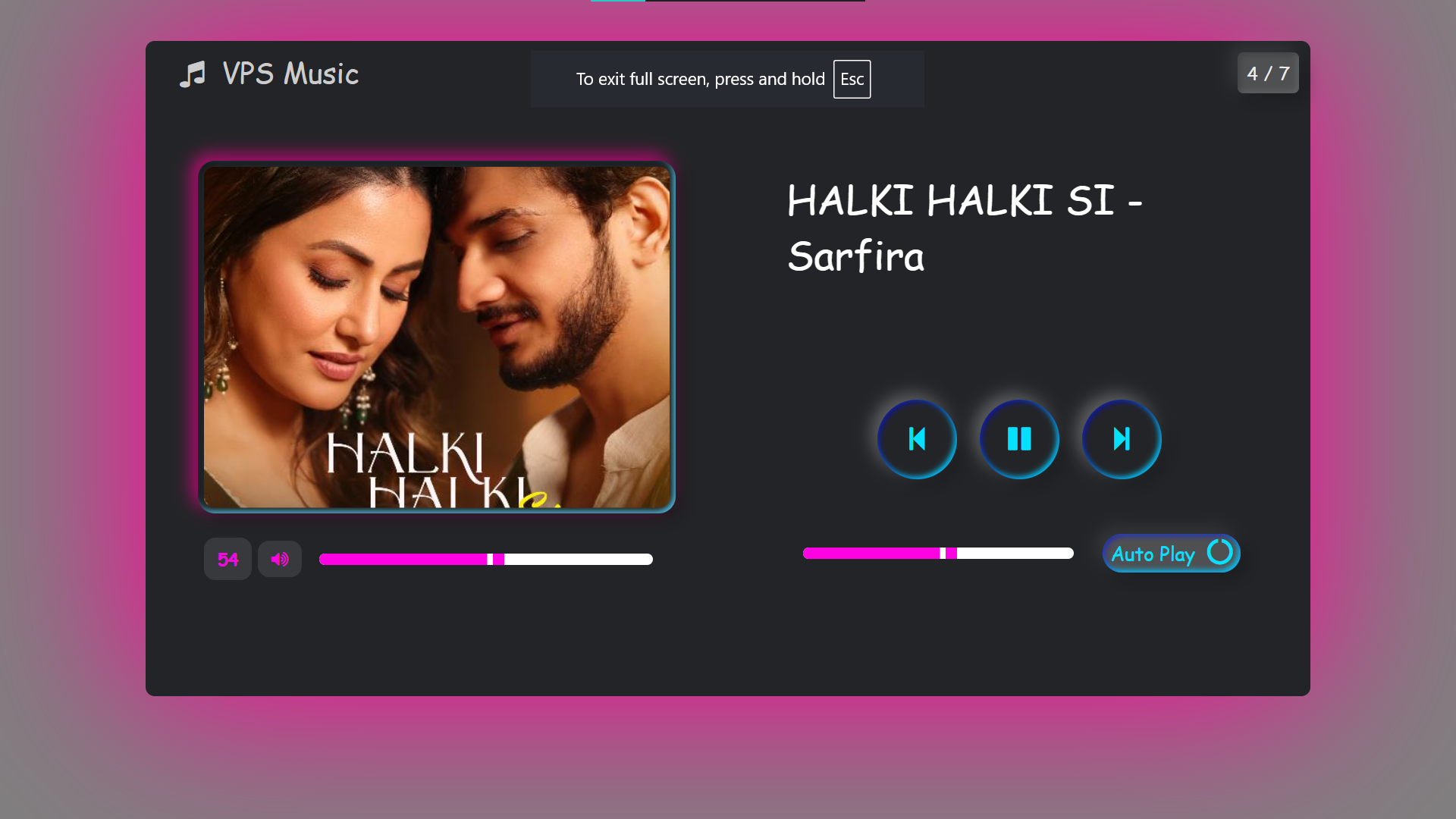
    play.innerHTML = '<i class="fa fa-play" aria-hidden="true"></i>';

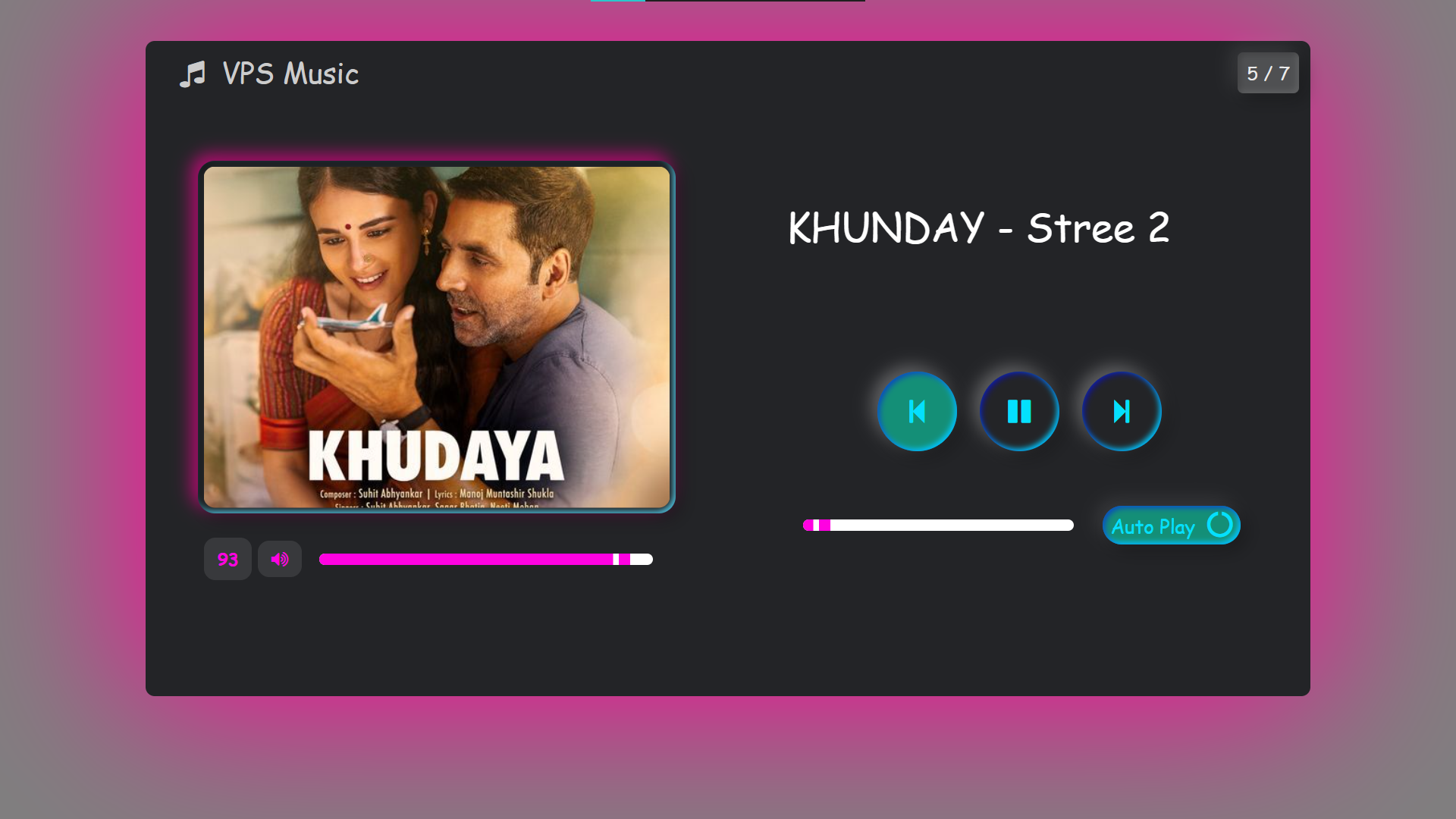
    if (autoplay == 1) {

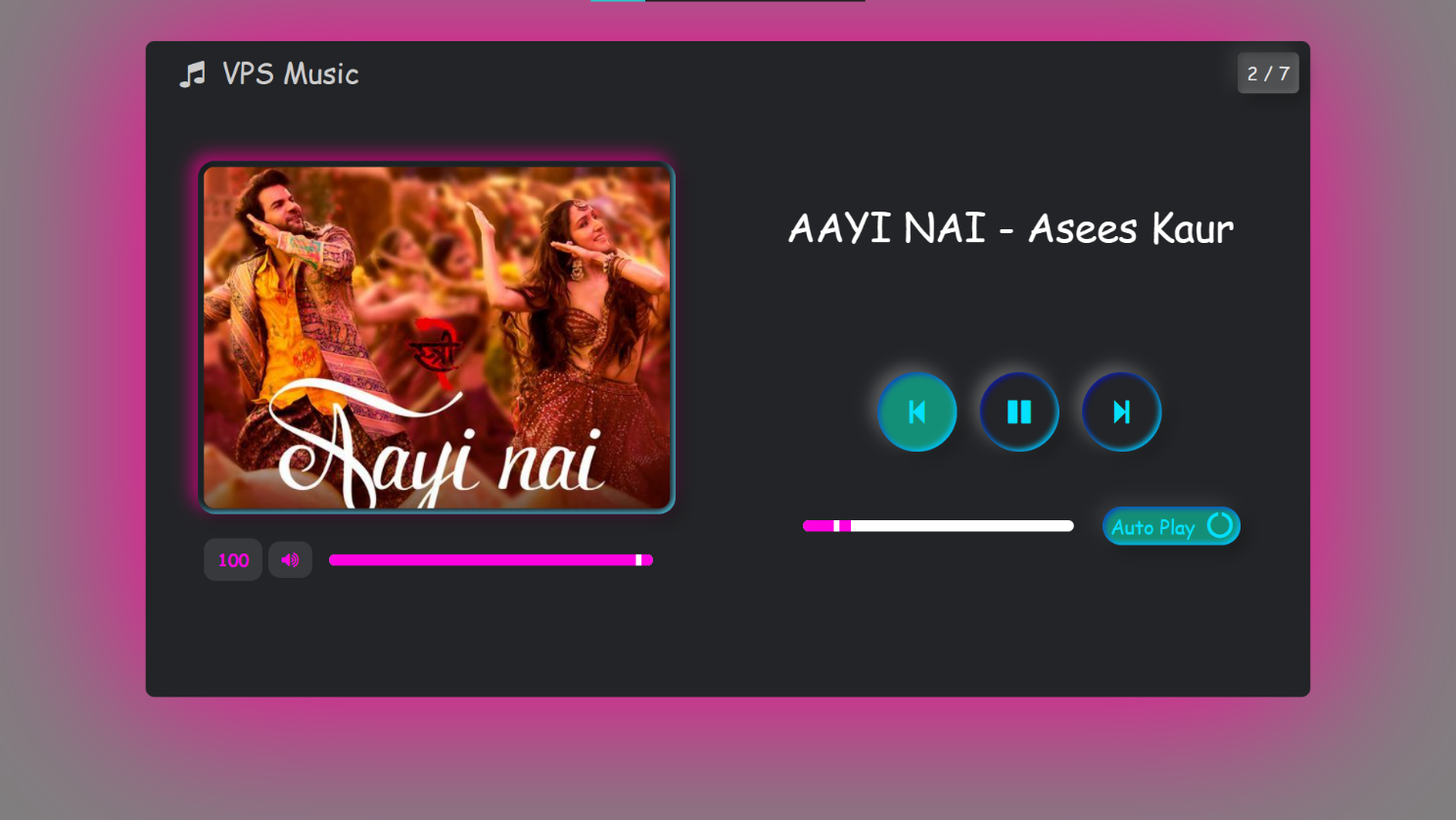
      index\_no += 1;

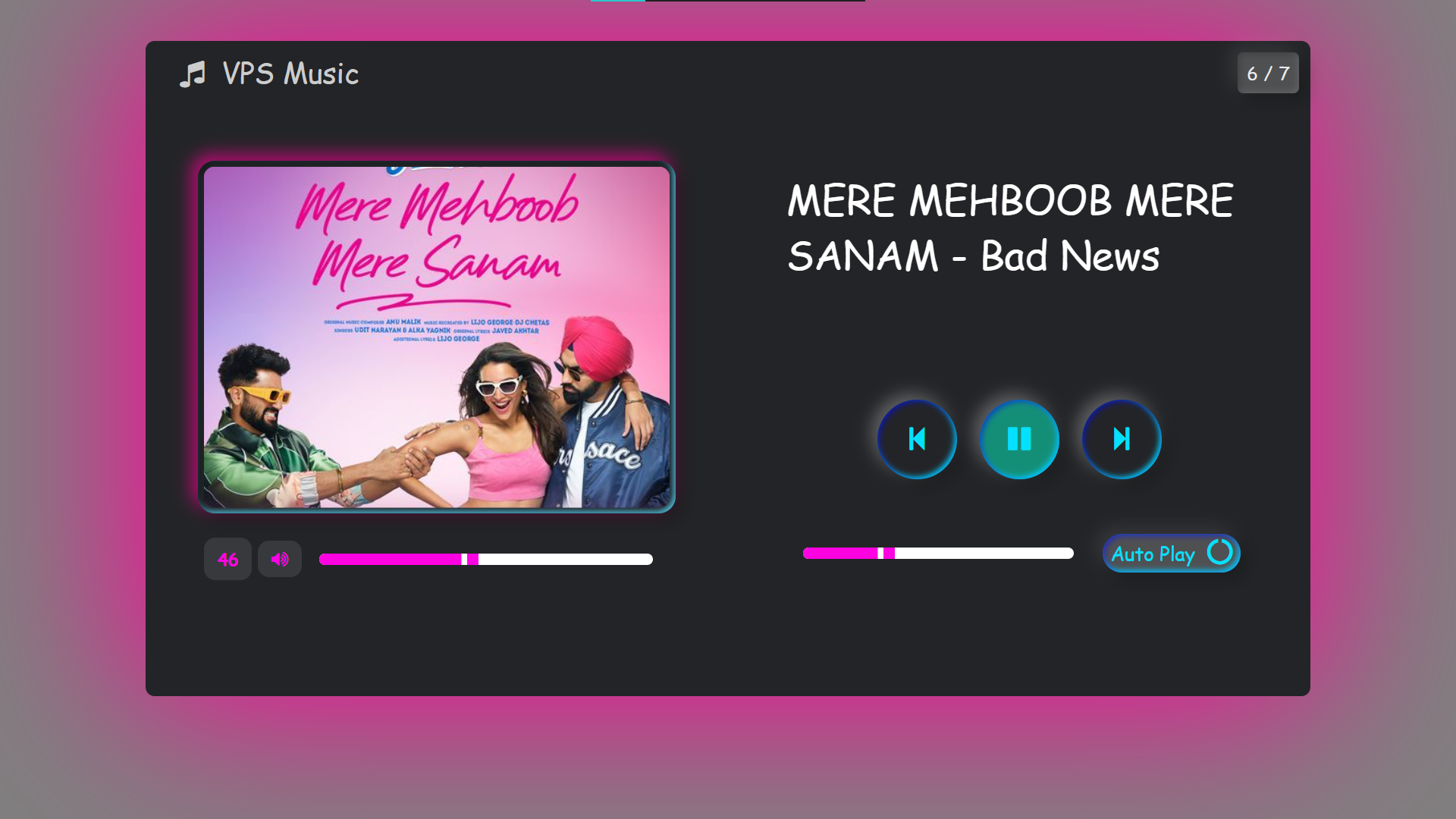
      load\_track(index\_no); playsong();}}}

* + - 1. Output :









* + - 1. Reference :

<https://chat.openai.com/chat> <https://www.geeksorgeeks.org/> <https://www.Googal.com/> https://www.github.com /

* + - 1. Resources Used :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. no** | **Instruments** | **Specifications** | **Quantity** | **Remark** |
| 01 | Computer system | Laptop I5, RAM 8 GB SSD 512 GB | 2. | ok |
| 02 | Software | VS Code | 1. | ok |
| 03 | Any other resources  use | MS Word  Chrome | 1. | ok |

* + - 1. Skill Development :

1. Communication skill
2. Leadership skill
3. Team work skill
4. Scientific approach development
5. Data collection skill
6. Research skill
   * + 1. Conclusion :

***Prof.Jondhale D.R Prof.Chaudhari N.K***

***(Project Guide) (H.O.D) IT***