**AUDIYO**

**A PROJECT REPORT SUBMITTED TO**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE**

**AWARD OF THE DEGREE OF**

**BACHELOR OF COMPUTER APPLICATIONS**

**BY**

**JORDAN GEORGE ABRAHAM**

**RA1931241010107**

**UNDER THE GUIDANCE OF**

**Mrs. R ANITA JASMINE, M.C.A., M.E.**

****

**DEPARTMENT OF COMPUTER APPLICATIONS**

**FACULTY OF SCIENCE AND HUMANITIES**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**Kattankulathur - 603 203**

**Chennai, Tamilnadu**

**April - 2022**

**BONAFIDE CERTIFICATE**

This is to certify that the project report entitled "**AUDIYO**" is the bonafide

work of **JORDAN GEORGE ABRAHAM**  (**RA1931241010107**) who carried out the project under my supervision for the award of Degree of Bachelor of Computer Applications. To my knowledge the work reported herein is the original work done by this student.

**Ms.R Anita Jasmine** **Dr. S. Albert Antony Raj**

Assistant Professor (Sr. G), Deputy Dean and Head,

Department of Computer Applications Department of Computer Applications

(GUIDE)

**Internal Examiner**  **External Examiner**

**ACKNOWLEDGEMENT**

With the profound gratitude to the **ALMIGHTY,** I take this chance to thank

the people who helped me to complete this project.

I take this as a right opportunity to say THANKS to my **parents** who are

there to stand with me always with the words "YOU CAN".

I am thankful to **Dr. T. R. Paarivendhar**, Chancellor, SRM Institute of Science and technology who gave me the platform to establish myself to reach greater heights.

I earnestly thank **Dr.A.Duraisamy**, Dean, Faculty of Science and Humanities, SRM Institute of Science & Technology who always encourage us to do novel things.

I express my sincere thanks to **Dr. S. Albert Antony Raj M.Sc., M.Phil., Ph.D.,** Deputy Dean, Department of Computer Applications,for his valuable guidance and support to execute all incline in learning.

It is my delight to thank my project guide **Ms.R.Anita Jasmine M.C.A., M.E.,**

Assistant Professor, Department of Computer Applications for his help, support,encouragement, suggestions, and guidance throughout the development phases of the project.

I convey my gratitude to all the faculty members of the department who extended their support through valuable comments and suggestions during the reviews.

A great note of gratitude to friends and people who are known and unknown

to me who helped in carrying out this project work a successful one.

**JORDAN GEORGE ABRAHAM**

**TABLE OF CONTENTS**

**Abstract**

1. **Introduction**

1.1 Problem Statement

1.2 Background Information and Motivation

1.3 Objectives

**2. Software Requirement Analysis**

2.1 Literature survey

2.2 Software Specification

2.3About the Software and its Features

2.3.1 HTML

2.2.2 CSS

2.2.3 JAVASCRIPT

2.2.4 VS Code

**3. System Analysis**

3.1 Existing System

3.2 Proposed System

3.3 Feasibility Study

3.3.1. The feasibility analysis

3.3.2. Economic feasibility

3.3.3. Technical feasibility

3.3.4. Social Feasibility

3.3.5. Saturation Overview

**4. System Design**

4.1 Use Case Diagram

4.2 Class Diagram

4.3 Activity Diagram

4.4 Sequence diagram

4.5 Component Diagram

4.6 ER Diagrams

**5. Coding**

5.1. Modules.

5.2 Code

**6.Testing**

6.1Testing

6.2 Report of testing.

**7. Output Screens**

7.1 Main Screen

7.2 Songs From Playlist

1. **Conclusion**
2. **Future Work**

**10. Bibliography**

**ABSTRACT**

This project is about the mp3 music player application development using Android. The biggest difference between the music player and existing applications is that it is completely free for users to use. It will integrate the advantages of existing music players on the market, as far as possible to mining out the existing music players' function, and then do the filtering in order to eliminate function that not practical or low cost-effective. Also, it will be keep improved based on user feedback.

In addition, depending on the user's usage scenario, the music player will also add some modes, such as driving mode and night mode, to allow users to use the application in any situation or environment. Moreover, the music player will have audio trim features, allowing users to trim the best part of their favorite song into phone ringtone or alarm. On the other hand, the existing music players pay less attention to the control of gestures. Therefore, the music player will solve the limitation by adding more gestures and shake the phone feature for media control to make it more user-friendly and humanity.

In a nutshell, the methodology for developing the mp3 music application used in this project is the agile development cycle. The agile development cycle consists of six phases, which is requirements analysis, planning, design, implementation or development, testing, and deployment. Due to the iterative and flexible nature of this approach, it is able to effectively adapt to users with changing requirements.

**CHAPTER 1**

**INTRODUCTION**

**Introduction**

“I don’t sing because I’m happy;

I’m happy because I sing.”

Research proves that when you listen to music you like, your brain releases dopamine, a “feel-good” neurotransmitter. Valorie Salimpoor, a neuroscientist at McGill University, injected eight music-lovers with a radioactive substance that binds to dopamine receptors after they listened to their favorite music. A PET scan showed that large amounts of dopamine were released, which biologically caused the participants to feel emotions like happiness, excitement, and joy.

So the next time you need an emotional boost, listen to your favorite tunes for 15 minutes. That’s all it takes to get a natural high!

* 1. **Problem Statement**

The problem domains on this project are:

1. **Bloated software and user interfaces**

Due to the fierce competition between music player applications, many developers tried to add many features, advertise and content to their respective music player in order to retain their users and attract new users. This trend has made it harder for users to get content from their music player, which also means it's harder to filter the content that they want. With the continuous iteration of application and a growing number of features, the music player will become even more bloated and the user's experience will become less smooth.

1. **Lack of gestures to control**

Most music player apps use touch buttons to play, pause and switch between previous and next songs while ignoring the convenience of using gesture swiping to control the music player. For instance, when a user is working and intends to skip to the next song in the music player, he/she have to switch their attention to the console from work and click the button. This problem does not affect music player properly work, but it does have some inconvenience.said that as our physical devices and appliances develop the button-free design, consumers will become more comfortable and confident in this way of interaction, so we should consider using gesture control on more mobile applications.

1. **Lack of sorting and searching features**

When users continuously to add new songs into the playlist, the difficulty of the songs the user wants to filter will increase. After the songs in the playlist are added to reach hundreds of songs, the user can only search song by continuously swipe up or down. If not carefully check the content, it is possible to miss the songs that the user wants to filter, and then repeat the behavior until the result is found. Therefore, it is an extremely poor experience for users.

**1.2 Background Information and Motivation**

In modern society, people live a fast-paced life, and pressure is constantly present in lives. Due to the wide use of mobile phones, music has become the daily essential spiritual food, everyone's mobile phone inside there must be a music player. An application like MP3 music players is used to balance stress and happiness. It accompanies people anytime, anywhere and anyplace such as when people taking the bus and exercising.

The mobile MP3 music player application is designed to allow users to listen to music in a more convenient and comfortable way without too much restriction. Moreover, it can play the music properly without interference from advertisements and offline.

Since many developers realize that modern urbanites are living in a stressful situation, they have captured the commercial opportunity, therefore many similar applications have emerged in the market. These applications have easy-to-use interfaces and features that make the user experience better.

However, these existing music players blindly pursue fancy appearance and huge features, resulting in the high utilization rate of users' mobile phones, such as CPU and memory. Whereas, for most normal users, these kind of huge and many features are meaningless. Therefore, this project is designed to dedicate to MP3 music player based on the Android mobile phone platform to optimize performance and simplify to meet user needs.

**1.3 Objectives**

The objective of this thesis is to propose development of android that:

1. **Make it with a simple feature and run smoothly**

By using this mp3 music player will make users fell comfortable and relaxed because it will pay more attention to the features commonly used by users, excluding some rarely used features that occupy a large of system processors, making the music player lightweight, simple, but also has powerful basic features.

1. **Support gesture control**

The MP3 music player will add features triggered by gestures to make it easier for users to use as well as less dependent on touch buttons. For example, a user can skip next or previous songs by simply swiping left and right on the anywhere of the screen in the playing interface.

1. **Support quick search**

The lack of a search bar in the music list is unacceptable. Therefore, the mp3 music player will use the search bar as well as fast scroll using alphabets on the right side of the screen, allowing users to quickly filter through hundreds of songs to find the ones users want to play.

**CHAPTER 2**

**SOFTWARE REQUIREMENT ANALYSIS**

**Software Requirement Analysis**

**2.1 Literature survey**

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then next steps is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for

developing the proposed system.

Since the commercialization of the web, web development has been a growing industry. The growth of this industry is being driven by businesses wishing to use their website to sell products and services to customers. web-development software, such as Adobe and Microsoft Visual Studio. An ever growing set of tools and technologies have helped developers build more dynamic

and interactive websites. Further, web developers now help to deliver applications as web services which were traditionally only available as applications on a desk-based computer. This has allowed for many opportunities to decentralize information and media distribution. Examples can be seen with the rise of cloud services such as Adobe Creative Cloud, Dropbox and Google Docs. These web services allow users to interact with applications from many locations, instead of being tied to a specific workstation for their application environment.

Examples of dramatic transformation in communication and commerce led by web development include e-commerce. Online auction-sites such as eBay have changed the way consumers find and purchase goods and services. Online retailers such as Amazon.com and Buy.com (among many others) have transformed the shopping and bargain-hunting experience for many consumers. Another good example of trans formative communication led by web development

is the blog.

**2.2 Software Specification**

**2.2.1 HTML ( Hypertext Markup Language)**

Html is open source language and all browser are support to html language .

Its use for create UI structure in website. Its main part of front-emd.



**2.2.2 JavaScript**

JavaScript is scripting language is support to the make HTML page make dynamic. JavaScript its second main part in front end to develop dynamic web page.



**2.2.3 CSS (Cascading Style Sheets)**

CSS is use for designing the HTML page. Using CSS HTML page user attract

Towards website.



**2.3About the Software and its Features**

**2.3.1 HTML**

The first public mention of HTML was in the year 1991 as Tim Berners-Lee described the 18 tags that were initially available in the markup language. It was 1994 when a working group was established for HTML - The Internet Engineering Task Force (IEFT). They believed the web belonged to the people (an idea that still holds strong today, especially with the battles over net neutrality), and so their doors were open to anyone who wished to join. By the year 1995 we were graced with many new tags and attributes, such as being able to set the font and background color. This drew some criticism as the argument was that HTML should only be used for describing the

structure of a document, and not for styling which went outside the scope of this. At this time, browser vendors were tearing ahead of the HTML working group, and finding consensus on new features came very slowly.

To tackle this, the vendors came together to dedicate themselves to standardizing HTML., and a month later the HTML working group was disbanded due to its slow-moving nature.

Ever remember slicing up a PSD, then firing up Dreamweaver to create a table to hold all those images? It's something we wouldn't dream of doing today, but oh how we still yearn for that level of simplicity. We've come a long way when looking at HTML 5. It no longer refers to just the tags, but also an API that lets us grab information such as glocalization, or a device's battery level. It's important to keep an eye on your own battery level when keeping up to date as well.

**2.2.2 CSS**

It wasn't long ago that all our styles were usually confined to a single CSS file.

We'd construct styles without giving them too much thought compared to the specificity minefield we know it as today. Sass (Syntactically Awesome Style-sheets) arrived in 2006, but it wasn't until a few years later when methodologies like OOCSS (Object-Oriented CSS) really took hold of the web for scalable, maintainable and re-usable styles. Nowadays we may use naming methodologies such as BEM (Block, Element, Modifier), or SUIT CSS, but what results is good old CSS in the browser. What's coming in the future looks promising with CSS variables, and hopefully element queries.

**2.2.3 JAVASCRIPT**

Of the 3 cornerstones of front-end development, JavaScript is the one that has undergone the most radical transformation. It saw daylight in the year 1995 after being in development for only 10 days. To this day you either love it, or hate it (we love it). Server-side JavaScript was also implemented in 1995, but the true champion of server-side JS is Node which was released in 2009.

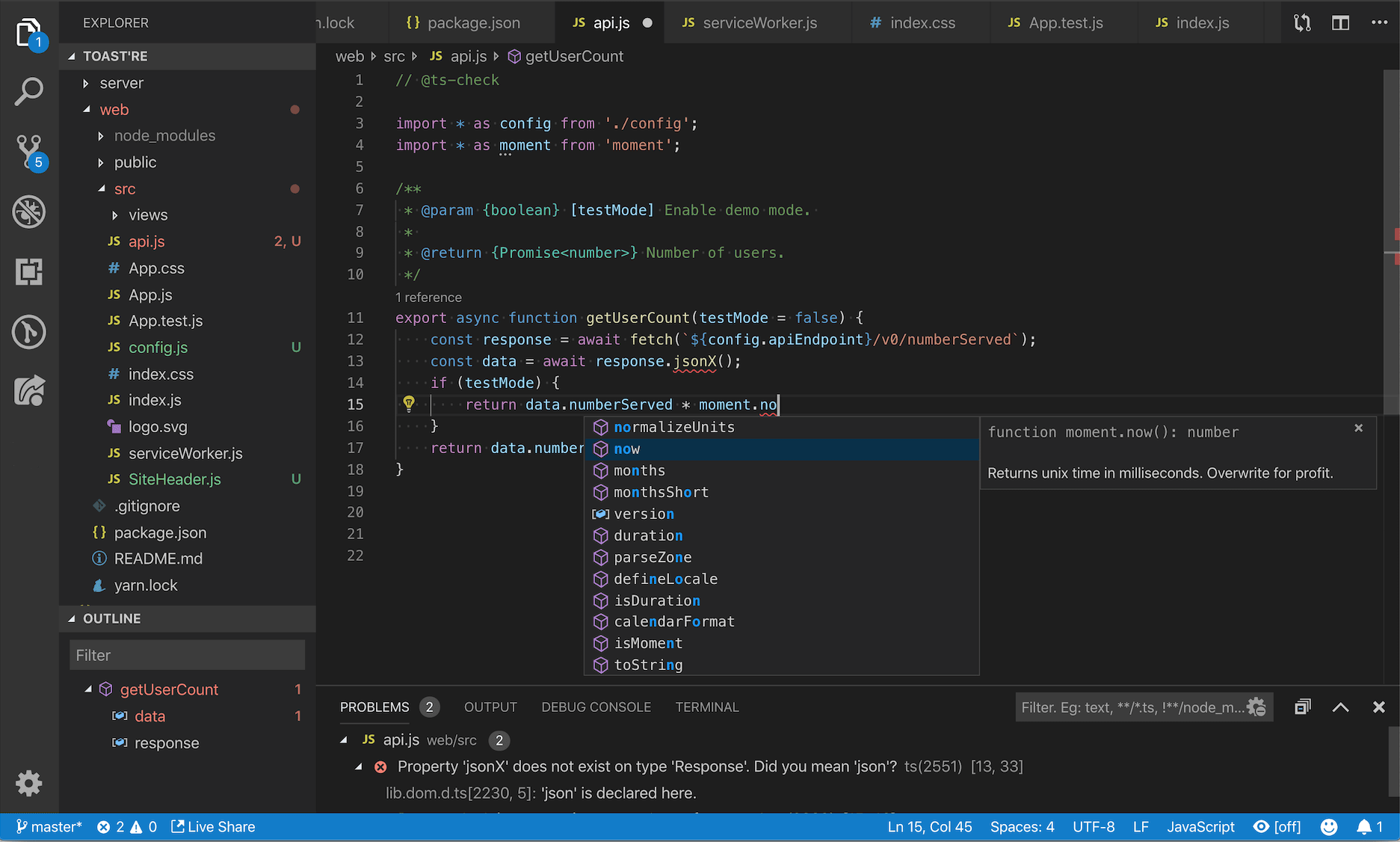
Since Node, it seems everything is running on JavaScript. Our build tools are using it, native apps utilize it... it has been embedded in almost every single aspect of development. JavaScript itself is growing stronger every year with a newly refreshed release cycle. ES2015 has been the biggest upgrade to the language recently, with yearly releases to follow suit. Thankfully these next releases won't be as far reaching as ES2015, which gives us a chance to catch up.

**2.2.4 VS Code**

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including [Java](https://en.wikipedia.org/wiki/Java_(programming_language)" \o "Java (programming language)), [JavaScript](https://en.wikipedia.org/wiki/JavaScript" \o "JavaScript), [Go](https://en.wikipedia.org/wiki/Go_(programming_language)" \o "Go (programming language)), [Node.js](https://en.wikipedia.org/wiki/Node.js" \o "Node.js), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)" \o "Python (programming language)) and [C++](https://en.wikipedia.org/wiki/C++" \o "C++).It is based on the [Electron](https://en.wikipedia.org/wiki/Electron_(software_framework)" \o "Electron (software framework)) framework, which is used to develop [Node.js](https://en.wikipedia.org/wiki/Node.js" \o "Node.js) [Web applications](https://en.wikipedia.org/wiki/Web_application" \o "Web application) that run on the [Blink layout engine](https://en.wikipedia.org/wiki/Blink_layout_engine" \o "Blink layout engine). Visual Studio Code employs the same editor component (codenamed "Monaco") used in [Azure DevOps](https://en.wikipedia.org/wiki/Azure_DevOps_Server" \o "Azure DevOps Server) (formerly called Visual Studio Online and Visual Studio Team Services).

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting" \o "Syntax highlighting), [bracket matching](https://en.wikipedia.org/wiki/Bracket_matching" \o "Bracket matching), [code folding](https://en.wikipedia.org/wiki/Code_folding" \o "Code folding), and configurable snippets. Visual Studio Code also ships with [IntelliSense](https://en.wikipedia.org/wiki/Intelligent_code_completion" \o "Intelligent code completion) for JavaScript, TypeScript, [JSON](https://en.wikipedia.org/wiki/JSON" \o "JSON), [CSS](https://en.wikipedia.org/wiki/CSS" \o "CSS), and [HTML](https://en.wikipedia.org/wiki/HTML" \o "HTML), as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspace for future reuse. This allows it to operate as a [language-agnostic](https://en.wikipedia.org/wiki/Language-agnostic" \o "Language-agnostic) code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.



**CHAPTER 3**

**SYSTEM ANALYSIS**

**SYSTEM ANALYSIS**

**3.1 Existing System**

Current music player required lot of internet to play songs. In low quality of internet is not run smoothly. When internet connection was stuck song is buffering and presentation of songs look very bad.For play songs we always go to page and play song the was not looping system. When playlist was over song song will stop to play.

Multiple songs are not available. When any music play half then you will not continued soon. You want play music return. Not user friendly option was there. Song will not download to system. And you can not upload song as our like. When you install app its get lot of space from device.

All system required to login and give access to your device. Song is not play on Dolby audio. Graphical User Interface (GUI) that is straightforward and easy to navigate has been designed. This GUI provide various screens with appropriate incorporate icons, hyperlinks etc. to facilitate screen navigation and data entry. The user can easily add items to their cart, and they can easily remove items form the cart if they needed. The user has the ability to return to home page from any location within the application.

**3.2 Proposed System**

Audiyo music system is build on JavaScript and HTML its have no any other support hence its website is so fast. In server and website no any middle technology used when you click on play button its start downloading song and play also. That why its not stuck the song during play. After The play of song downloaded song its also deleted from browser its help to take low data storage.

You can Upload and download song also. Its so user friendly and use less internet data. Only those think its fetch its required. Its depends upon priority which data is required and what operation perform. Hence its smooth than other application

No need to sing In the data you directly play song. Its Application not required to download and install. All new song available on music player Provide a dolby sound system its very amazing to listen music. So much clarity on play song.

**3.3 Feasibility Study**

**3.3.1. The feasibility analysis:**

This section verified that it is feasible to add music player on the Android system from the aspects of economic, technical and social feasibility.

**3.3.2. 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.

**3.3.3. Technical feasibility:**

To design a music player which meets the basic requirements, a deep understand of JAVA language, the Android system architecture, 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.

**3.3.4. 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.

**3.3.5. Saturation Overview.**

This section describes requirements of the system based on basic control functions of players, list management of the player and system setup function of the player according to research results of the project demand.

According to the research results of project demand, the basic requirements of project system and its function structure are presented. And describe the demand of the system through the different angles. The project is divided into the following parts by using diagram: the basic control functions of the player, the playlist management function of the player and system setting function of the player. The player interface requires rational layout, comfortable color, friendly control buttons and concise and beautiful images. According to the Android system requires, the non- response time is 5 seconds. The following is requirements in the music player development application.

Application response time shall not exceed 5 seconds in music playing.

Application response time shall not exceed 5 seconds as the music is suspended.

Application response time shall not exceed 5 seconds as the music is stopped.

Application response time shall not exceed 5 seconds as Move Next/Move Previous music.

Application response time shall not exceed 5 seconds as system listing is required.

**CHAPTER 4**

**SYSTEM DESIGN**

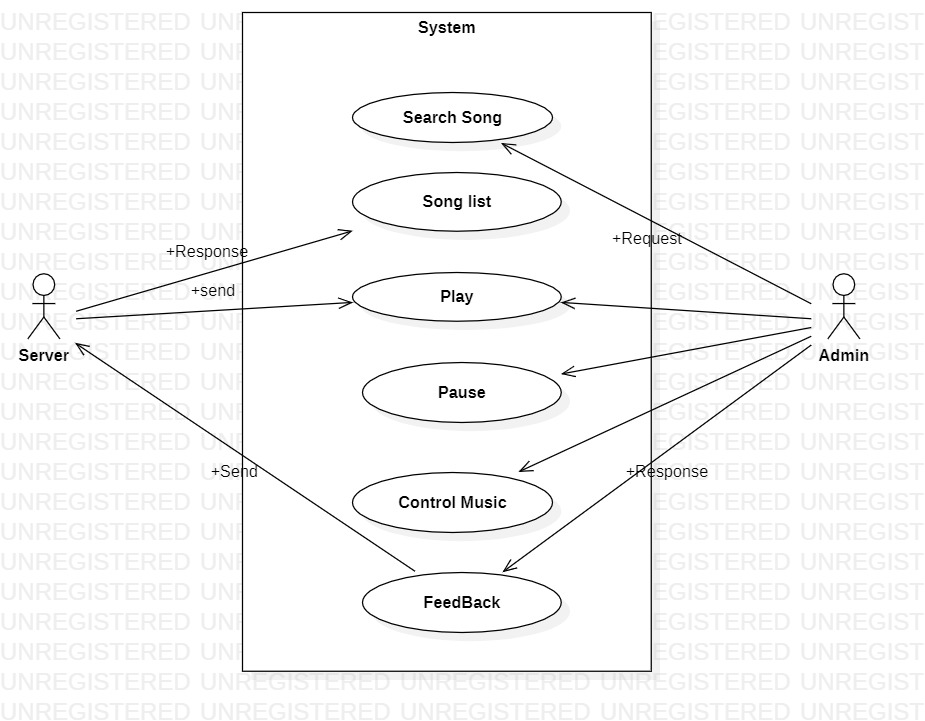
**System Design**

1. **UML diagrams : -**

A diagram is the graphical presentation of a set of elements, most often rendered as a connected graph of vertices and ares. You draw diagram to visualize a system from different perspective, so a diagram is a projection into a system. For all but most trivial systems, a diagram represents an elided view of the elements that make up a system. The same element may appear in all diagrams, only a few diagrams, or in no diagrams at all. In theory, a diagram may contain any combination of things and relationships, In practice, however, a small number of common combinations arise, which are consistent with the five most useful views that comprise the architecture of a software-intensive system. For this reason, the UML includes nine such diagrams:

**4.1 Use Case Diagram:**

A use case diagram in the Unified Modeling Language (UML) is atype of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.



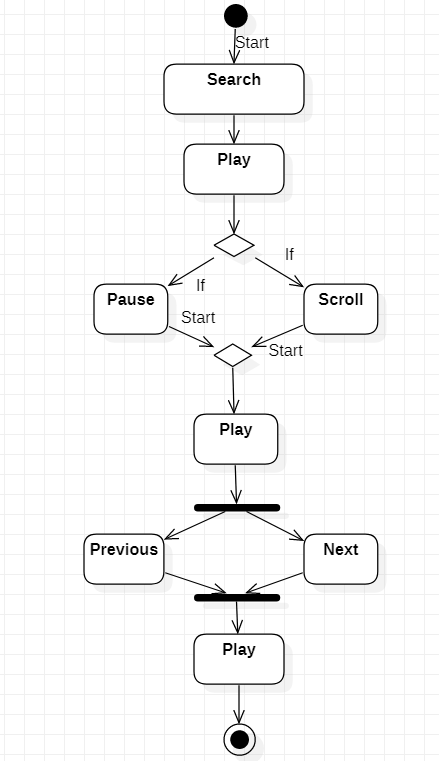
**4.2 Class Diagram**

The **[UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language" \t "https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/_blank)** Class diagram is a graphical notation used to construct and visualize object oriented systems. A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's:



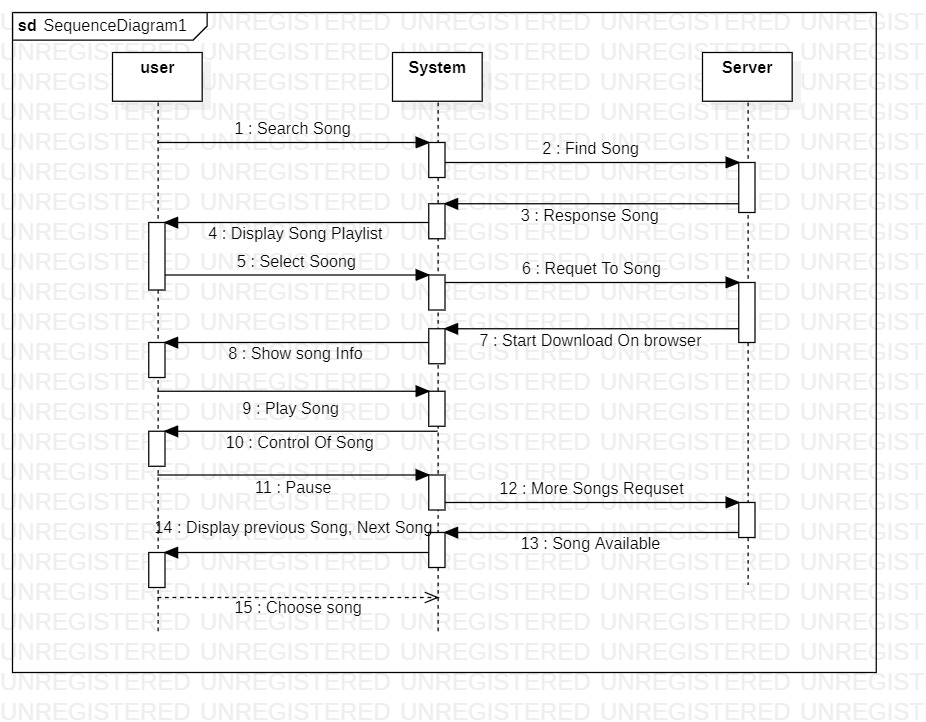
**4.3 Activity Diagram :**

We use Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.



**4.4 Sequence diagram**

Sequence diagrams are a popular dynamic modeling solution in UML because they specifically focus on lifelines, or the processes and objects that live simultaneously, and the messages exchanged between them to perform a function before the lifeline ends. Along with our [UML diagramming tool](https://www.lucidchart.com/pages/examples/uml_diagram_tool" \t "https://www.lucidchart.com/pages/_blank), use this guide to learn everything there is to know about sequence diagrams in UML.

****

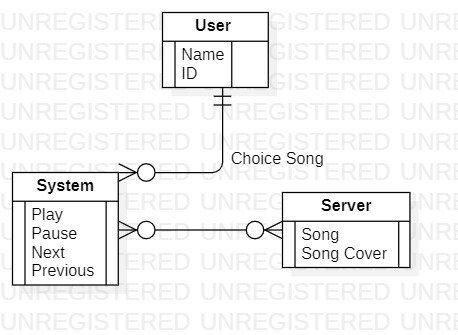
**4.5 Component Diagram**

[UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language" \t "https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/_blank) Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.



**4.6 ER Diagrams**

****ER Diagram**** stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.



**CHAPTER 5**

**CODING**

**Coding**

**5.1. Modules.**

**5.1.1. Play:**

Play song modules is use for playing songs. When click any song this song start playing on click on song. At down bar we also option of playing song.

**5.1.2 Pause :**

Pause is use for stop the song where song is play now. Its pause the song at the time of song. When click return its start song from those time where song would pause.

**5.1.3 scroll bar :**

When wanted to lay song from any minutes later scroll the bar on this time. The song will play.

**5.1.4 Next Song :**

When click on next song button . The Next song will play from playlist.

**5.15 Previous song :**

When click on previous button . The previous song play from playlist.

**5.2 Code**

**Index . html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Audiyo- Listen without limit</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <nav>

        <ul>

            <li class="brand"><img src="logo.png" alt="Audiyo">Audiyo</li>

            <li>Feedback</li>

            <li>About</li>

        </ul>

    </nav>

    <div class="container">

        <div class="songList">

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

        <div class="songItemContainer">

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="0" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="1" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="2" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="3" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="4" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="5" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="6" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

            <div class="songItem">

                <img alt="1">

                <span class="songName">Fall in Love</span>

                <span class="songlistplay"><span class="timestamp">02:36 <i id="7" class="fa-solid songItemPlay fa-circle-play"></i></span></span>

            </div>

        </div>

        </div>

        <div class="songBanner"></div>

    </div>

        <div class="bottom">

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

            <div class="icons">

                <!-- fontawesome icons -->

                <i class="fa-solid fa-2x fa-backward-step" id="previous"></i>

                <i class="fa-solid fa-2x fa-circle-play" id="masterPlay"></i>

                <i class="fa-solid fa-2x fa-forward-step" id="next"></i>

            </div>

            <div class="songInfo">

                <img src="playing.gif" width="45px" alt="" id="gif"> <span id="masterSongName">Fall in Love</span>

            </div>

        </div>

    </div>

    <script src="https://kit.fontawesome.com/8ddd406db6.js" crossorigin="anonymous"></script>

    <script src="script.js"></script>

</body>

</html>

**Script.js**

console.log("Welcome to Audiyo");

//Initialize the Variables

let songIndex= 0;

let audioElement = new Audio('songs/1.mp3');

let masterPlay = document.getElementById('masterPlay');

let myProgressBar = document.getElementById('myProgressBar');

let gif = document.getElementById('gif');

let masterSongName = document.getElementById('masterSongName');

let songItems = Array.from(document.getElementsByClassName("songItem"));

let songs = [

    {songName: "Fall in Love", filePath: "songs/1.mp3", coverPath: "covers/1.jpg"},

    {songName: "Heroes Tonight", filePath: "songs/2.mp3", coverPath: "covers/2.jpg"},

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

    {songName: "My Heart", filePath: "songs/4.mp3", coverPath: "covers/4.jpg"},

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

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

    {songName: "Sky High", filePath: "songs/7.mp3", coverPath: "covers/7.jpg"},

    {songName: "Symbolism", filePath: "songs/8.mp3", coverPath: "covers/8.jpg"},

]

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

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

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

})

// audioElement.play();

// Handle play/pause click

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

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

        audioElement.play();

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

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

        gif.style.opacity = 1;

    }

    else{

        audioElement.pause();

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

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

        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;

})

const makeAllPlays = ()=>{

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

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

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

    })

}

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

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

makeAllPlays();

songIndex = parseInt(e.target.id);

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

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

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

masterSongName.innerText = songs[songIndex].songName;

audioElement.currentTime = 0;

audioElement.play();

gif.style.opacity = 1;

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

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

    })

})

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

    if(songIndex>=7){

        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-circle-play');

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

})

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-circle-play');

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

})

**Style.css**

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

@import url('https://fonts.googleapis.com/css2?family=Roboto+Slab:wght@300&family=Ubuntu&display=swap');

body{

    background-color:antiquewhite;

}

\*{

    margin: 0;

    padding: 0;

}

nav{

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

}

nav ul{

    display: flex;

    align-items: center;

    list-style: none;

    height: 70px;

    background-color: black;

    color: white;

}

nav ul li{

    padding: 0 15px;

}

.brand img{

    width: 87px;

}

.brand{

    display: flex;

    align-items: center;

    font-weight: bolder;

    font-style:oblique;

font-size: 1.3rem;

}

.container{

    min-height: 67vh;

    background-color: black;

    color: white;

    font-family: 'Roboto Slab', sans-serif;

    display: flex;

    margin: 25px auto;

    width: 70%;

    border-radius: 10px;

    padding: 30px;

    background-image: url('bg.jpg');

}

.bottom{

    position: sticky;

    height: 93px;

    background-color: black;

    color: white;

    bottom: 0;

    display: flex;

    justify-content: center;

    align-items: center;

    flex-direction: column;

}

.icons{

    margin-top: 10px;

}

.icons i{

    cursor: pointer;

}

#myProgressBar{

    width: 65vw;

    text-align: center;

    cursor: pointer;

}

.songItemContainer{

margin-top: 40px;

}

.songItem{

height: 45px;

display: flex;

background-color: white;

width: 90%;

color: black;

margin: 12px 0;

justify-content: space-between;

align-items: center;

border-radius: 30px;

}

.songItem img{

    width: 40px;

    margin: 0 20px;

    border-radius: 30px;

}

.timestamp{

    margin: 0 25px;

}

.timestamp i{

    cursor: pointer;

}

.songInfo{

    position: absolute;

    left: 0;

    font-family: 'Roboto Slab', sans-serif;

}

.songInfo img{

    opacity: 0;

    transition: opacity 0.4s ease-in;

  }

**CHAPTER 6**

**TESTING**

**Testing**

**6.1 Testing**

Testing is important part of any application. We are tested 10 songs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Song Name** | **Play**  **Button** | **Pause**  **Button** | **Previous**  **Button** | **Next**  **Button** | **Scroll**  **Bar** | **Dolby Audio** |
| **Soft Bass**  **Song 1** | Pass | Pass | Pass | Pass | Pass | Pass |
| **Soft Bass**  **Song 2** | Pass | Pass | Pass | Pass | Fail | Pass |
| **Soft Bass**  **Song 3** | Pass | Pass | Pass | Pass | Pass | Pass |
| **Heavy Bass**  **Song 4** | Pass | Pass | Fail | Pass | Pass | Pass |
| **Heavy Bass**  **Song 5** | Pass | Pass | Pass | Fail | Pass | Fail |
| **Heavy Bass**  **Song 6** | Pass | Pass | Fail | Pass | Pass | Pass |
| **Modern**  **Bass**  **Song 7** | Pass | Pass | Pass | Pass | Pass | Pass |
| **Classic**  **Bass**  **Song 8** | Pass | Pass | Pass | Pass | Fail | Pass |
| **Classic**  **Bass**  **Song 9** | Pass | Pass | Pass | Pass | Pass | Pass |
| **Classic**  **Song 10** | Pass | Pass | Pass | Fail | Pass | Pass |

**6.2 Report of testing.**

We are selected 10 Different song for testing ( Soft Bass, Heavy bass, Classical Song, Modern song ).

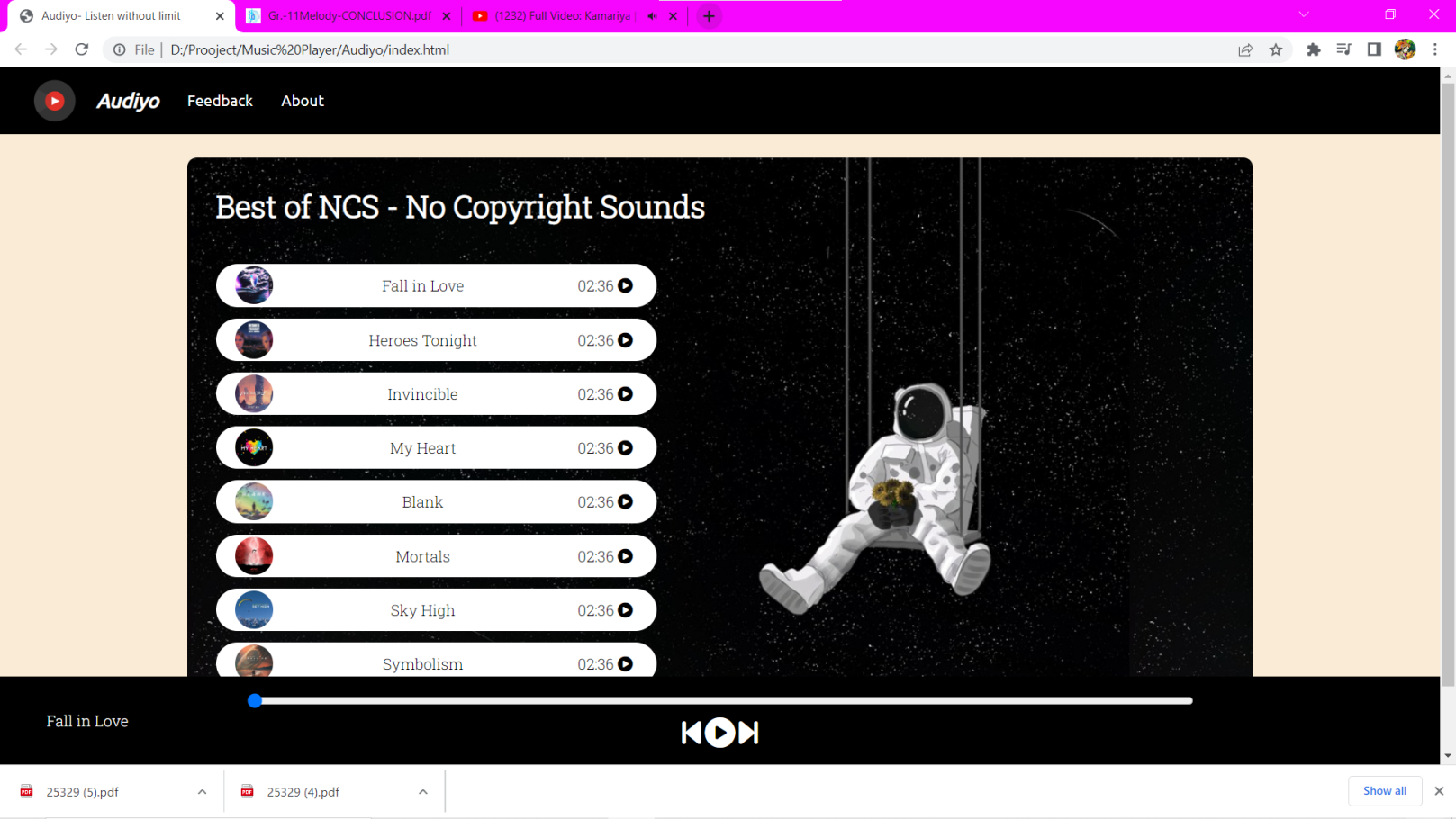
From this song Play and Pause button is perfectly Action. Previous and Next button failed some time. Hence don some change in code. All song having Dolby sound only one song is failed because of song is not perfect hence we think don feature development features for that type of song.

**CHAPTER 7**

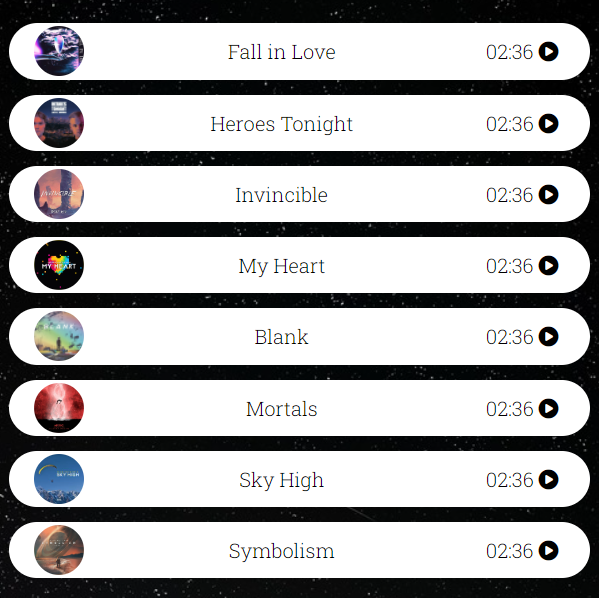
**OUTPUT SCREENS**

**Output Screens**

**7.1 Main Screen**



**7.2 Songs From Playlist**



**CHAPTER 8**

**CONCLUSION**

**Conclusion**

Through the development of music player on Web platform, we get a clear

understanding of overall process of the system. The core part of the music player is mainly composed of main interface, playlists, menus, play Settings, file browsing and song search. Grasping the development of the six parts, the music player has had the preliminary scale. Based on the function of the six categories, add some other small features.

Music player system realized the basic function of player: play, pause, and stop, up/down a, volume adjustment, lyrics display, play mode, song search, file browser, playlists query, and other functions. This development implicated the popular mobile terminal development technology. on file. The system realized the music player programming.

This design of music player based on Web system requires elaborate design of the music player framework, by adopting javscript language as design support of this system, with the css tools, and combination of HTML5 version lead to the comprehensive and smoothly design and development of the mobile terminal.

In a nutshell, when users hold the mentality of venting and relaxation to expect the music player to bring them relief pressure, in result the application with a dazzling and complex interface, a variety of multifarious functions, from time to time prompt out of the advertising, as well as the function that requires be a members to use, which will only make users feel more depressed and feel the pressure. Moreover, most people who use a music player, usually don't leave the music player open in the foreground, but start playing music and then go on to do something else at hand such as take a break, read a book and news, or play a game.

As a result, they can't focus on the various functions and buttons in the app's interface. For instance, users who are lying down to take a break and tried to switch to the next song but they need lots of action like unlocking the phone, open the app again in the background and look for the switch button. In addition, the specific song is overwhelmed by a large number of songs and cause information overload, users can only spend more energy and time to find it. For example, searching for a book in the library, and realize that there is no library catalog is mean to looking for a needle in a haystack.

In short, the proposed application will combine the strengths of most music players on the existing market and eliminate some unrealistic features, allowing users to focus on listening to music rather than store, communities or various VIP packages or features. The proposed MP3 music player will focus on improving the experience of users of the music player experience.

**CHAPTER 9**

**FUTURE ENHANCEMENT**

**Further Enhancements**

**9.1 Future Work : -**

1. Enhanced interactivity, allowing users to open song playlist when they swipe up from the music playing interface
2. Implement of the in-app download of songs, rather than the current use of a specific website as a hyperlink
3. Refactoring code, rebuild the coding structure to make the coding look cleaner, easier to understand and perform efficiently.
4. Cross-platform, running the app on IOS, not only Android

**CHAPTER 10**

**BIBLIOGRAPHY**

**Bibliography**

**10.1 Bibliography**

1. **<http://eprints.utar.edu.my/3793/1/16ACB05288_FYP.pdf>**
2. **<https://youtu.be/ANzPM5-lwXc>**
3. **<https://www.daitm.org.in/wp-content/uploads/2019/04/Gr.-11Melody-CONCLUSION.pdf>**

1. **<https://www.freeprojectz.com/premium-synopsis/synopsis-online-music-portal>**