GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

WEB MUSIC PLAYER

A Report for the ETE of Project 2

Submitted by

PIYUSH CHAUHAN

(1613101479/16SCSE101080)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION OF CLOUD COMPUTING AND VIRTUALIZATION

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING,

Under the Supervision of

Mr. Vivek Anand M., Research Scholar (Ph.D), Asst., Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"WEB MUSIC PLAYER"** is the bonafide work of "**PIYUSH CHAUHAN(1613101479)**" who carried out the project work under my supervision.

SIGNATURE OF HEAD
Dr. Raju Shanmugam
Ph.D. (CS),M.E
Professor & Dean, School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR
Mr. Vivek Anand M. **,**
Research Scholar(Ph.D.),
Professor,
School of Computing Science & Engineering

TABLE OF CONTENTS

CHAPTER NO. **TITLE**

ABSTRACT

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

INTRODUCTION

Music has always been a means of entertaining people even from the earliest ages of the civilization. Historically it was produced by musicians and only available during life concerts. The technological evolution made it possible to save the music on vinyl plates, later electromagnetic charged stripes, CDs until the technology brought us to saving tracks digitally. When dealing with a huge collection of tracks, people encounter management problems they did not have before.So they have to develop new ways of using the music collection for their entertainment. Playlists are a good approach for saving successions of tracks that one likes. The most dominant problem of existing playlist generation mechanisms is, however, their lack of flexibility: new tracks are not automatically added, they don't adapt to the user's current mood etc. A new approach in dynamically organizing tracks into playlists is on its way: companies like last.fm already suggest an algorithm of mapping songs one to each other based on their "similarities"; but how to compute these similarities? One way, that did not prove to be very productive, is to analyze the audio content of the track – its audio frequencies. This way, tracks are split in categories like "Heavy Metal" and "Blues", but people do not like all tracks of a certain gender and these genders might be inaccurate. Another way, which is given more and more

attention by researchers and companies worldwide, is computing similarities between tracks based on user input. As an example: if two users add the same two tracks to their playlists, one can deduce that these tracks are similar and so, also other people that pick one of them are likely to enjoy the other one as well.

The music player allows a user to play various media file formats. It can be used to play audio as well as video files. The music player is a software project supporting all known media files and has the ability to play them with ease.

<u>The project features are as follows:</u>

- User may attach Folder to Play add various media files within it.
- User may see track lists and play desired ones accordingly.
- Supports various music formats including .mp3, WMA, WAV etc.
- Interactive GUI.
- Consists of Pause/Play/Stop Features
- Consists of a Volume controller
- The system also consists of a sound Equalizer
- It Displays the media playing time with Track Bar so that user may drag the media play as needed.

Features:

- Unlimited song playlist list wise
- Easy customization via HTML / CSS files
- Supports multiple instances into a page
- Previous/Next, Play/Pause, Stop buttons
- Load new playlist
- Set volume

Merits:

- Enhanced audience experience.
- Opportunity to use new technology and special effects.
- easier to explain or put things in perspective
- helps keep mistakes at a minimum

Demerits:

- requires a well-designed presentation or material
- participants might pay more attention to the graphics than the audio
- Expense of equipment needed and expertise needed to set up and operate.
- Preparation and planning time needed.

Existing System

This chapter debates the necessity of a new solution to make the world of intelligent track comparison even more accessible to the end user,followed byan app roach sketch.

Motivation

Several solutions already use intelligent playlists embedded in music players installed on computers. There are also online solutions, the most popular of which islast.fm, which acts as a personalized radio station that plays preferre dmusic. On the other hand it does not allow playback of a certain track. There are also other solutions, like the genius function of iTunes or the Music Explorer; both use the user's music collection to generate playlists. The biggest disadvantage of the latter solution is that the user can use only tracks that he/she already has on his/her PC to generate playlists. Of course this limits the power or the algorithm very much. There are already services that provide the music content (like last.fm or YouTube to name a few) so it's a natural conclusion to try to use these service s in connection with the playlist-generating algorithm. In order to understand the utility of such an application, just imagine the following scenario: one enjoys listening to music while working. It is not

common to store music on the company's computer so one rather has a person al mp3 player with himself during office time. If one takes enough time to prep are ones playlists in order to fit ones current mood, it is a pretty decent soluti on. But what if new tracks appear that one might like? One first has to do serious research in order to find them and then go through buying them, downloading them to his/her mp3 player, updating the playlists it already sounds very difficult, right? Now the suggested scenario is the following: one opens a web site, types in a track that reflects ones current moo d and hits "play". That's it! The player chooses tracks that one likes, also plays new tracks that one did not hear before, and can go like this for hours and ho urs without repetition. One can go on with one's work and in order to stop the music, one only has to hit stop or close the browser. The simplicity of the sol ution speaks for itself. The goal of this thesis is to analyze and implement an a pproach of building such a web-based music player. The questions it has to find answers for are: How should the user interaction be designed to maximize the user satisfaction? Where to source the audi data from, while ensuring a maximum coverage? And finally, how to promote the applicatio n in order to attract as many users as possible? The different implementation p ossibilities are evaluated and the best solution is implemented. The logic behind the web-based music player computes a 5 sequence of tracks based on their similarity. At the same time, user behavioral data is gathered that

helps further releases to be even more user friendly. Another important aspect of the application is its extensibility. Modularity and code reusing are very important parameters of this application, as it acts as a version 1.0 for future releases. These future releases will be able to interact wit h the user for finding the best track video on YouTube or to determine the mu sic preferences of users and even adapt the space to the new usage statistics.

Approach

The analysis of the currently available tools to accomplish the task is one of the most important steps because the ground concepts of the application should never change, regardless of its future complexity. The several possible implementations of the web service together with the balancing of computing tasks between server and client are the first parameters that have to be defined for a solid base. Also the programming language plays a crucial part in the development process, as it is shown later. The amount of callbacks to the database in favor of less memory usage is also an important aspect that is difficult to estimate from the start. In order to allow a high flexibility while still maintaining a small dataflow, the implementation of the logic is mainly on the server. The UI responsibility is fully retained by the client side as well as servicing UI requests and only notify

the server of such activity. In order to achieve the high goals that were set, th e structure of the applicationis important to be highly modularized to allow interchanging the modules with better, more complex implementations. It is important to determine which components are possible and also easy to mod ularize, without introducing too much communication overhead in the interfaces. It turns out that the music content related jobs can easily be modula rized, as well as the DB relatedjobs and the playlist computing tasks. The core of the application only needs to handle these modules and the logging task. Also the communication with the client is modularized, making it particularly easy to implement new clients running on the same servic e or new services to servethe same client. .

Proposed Model

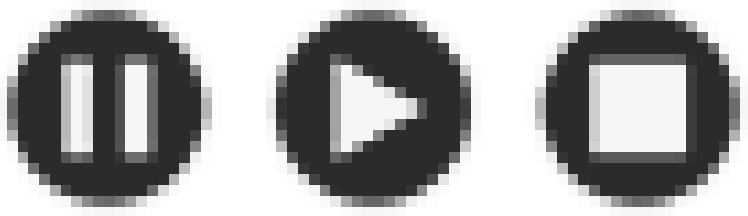
- The application is a simple HTML file that you open in your browser.
- You only need to download our zip file from the button near the beginning of the article, and unzip it somewhere on your computer.
- Unfortunately, due to security restrictions in modern browsers it won't work if you just double click the index.html file.

- You will have to open it through a locally running web server like Apache or Nginx and access it through local host. Or you can just use our demo, nothing is uploaded so your music is safe.
- The app listens for JavaScript drag and drop events.
- When you drop a mp3 file, it extracts information like song and artist name, if they are available, from the file's ID3 tags.
- Each song is placed in an array, which represents our playlist.
- The application then initializes the Wavesurfer.js audio player, which generates the awesome wave visualization for every song and plays it.
- From there on we can do everything you would expect from a native audio player - play next/previous, pause, pick songs and so on.
- Our playlist section also gives users the option to remove songs from the player or search for a particular track, album or artist.

Implementation/Work/Design in Progress

- The pop up effect for the playlist and other highlights and small animations were done via CSS by manipulating classes with jQuery.

- All of the icons we needed for this music player were already available in:



JavaScript Library for HTML Audio – Sound.js

- A JavaScript library that provides a simple API, and powerful features to make working with audio a breeze.

Audio let

- Audio let is a JavaScript library for real-time audio synthesis and composition from within the browser.

Open Source Audio Library **– Wedge.JS**

- Wedge.js is a small audio library that provides HTML5 apps with low latency audio if available, and falls back to Buzz! If not.

ION.Sound**– jQuery**Plugin for Playing Sounds on Events

- A free jQuery-plugin for playing sounds on events.

The HTML Audio Compatibility Layer **– Audio5js**

- Audio5js is a Javascript library that provides a seamless compatibility layer to the HTML5 Audio playback API, with multiple codec support and a Flash-based MP3 playback fallback for older or unsupported browsers.

- The motivation for creating Audio5js is to provide a light-weight, library-agnostic, Javascript-only interface for audio playback in the browser.

FUNCTIONS

```
// Progress bar
(function () {
    var progressDiv = document.querySelector('#progress-bar');
    var progressBar = progressDiv.querySelector('.progress-bar');

    var showProgress = function (percent) {
        progressDiv.style.display = 'block';
        progressBar.style.width = percent + '%';
    };

    var hideProgress = function () {
        progressDiv.style.display = 'none';
    };

    wavesurfer.on('loading', showProgress);
    wavesurfer.on('ready', hideProgress);
    wavesurfer.on('destroy', hideProgress);
    wavesurfer.on('error', hideProgress);
}());
```

```php
<?php
include_once dirname(__FILE__)."/libraries/getid3/getid3.php";
// Initialize getID3 engine
$getID3 = new getID3;

// Analyze file and store returned data in $ThisFileInfo
$file = @$_GET['file'];
if($file != '')
{
$ThisFileInfo = $getID3->analyze($file);

if(isset($ThisFileInfo['tags']['id3v1']))
echo json_encode($ThisFileInfo['tags']['id3v1']);
else if(isset($ThisFileInfo['tags']['id3v2']))
echo json_encode($ThisFileInfo['tags']['id3v2']);
else if(isset($ThisFileInfo['tags']['id3v3']))
echo json_encode($ThisFileInfo['tags']['id3v3']);
else
echo "{}";
}
?>
```

```html
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, minimum-scale=1.0, maximum-scale=1.0">
<script type="text/javascript" src="js/jquery.min.js"></script>
<script type="text/javascript" src="js/jquery.custom-scrollbar.js"></script>
<script type="text/javascript" src="js/wavesurfer.min.js"></script>
<script type="text/javascript" src="js/equalizer.js"></script>
<link rel="stylesheet" type="text/css" href="css/style.css">
<link rel="stylesheet" type="text/css" href="css/jquery.custom-scrollbar.css">


<title>My Player</title>

<script type="text/javascript">
$(document).ready(function(e) {
    $('#playlist').customScrollbar({updateOnWindowResize:true});
});

</script>
</head>

<body>


<div id="demo">
<div class="section group">
    <div class="col span_1_of_3">
        <div id="equalizer">
            <!-- Here be equalizer sliders -->
        </div>
    </div>
    <div class="col span_1_of_3">
        <div id="info-wrapper">
```

Follow these steps to add sound to a Web page:

- Open your Web page in Notepad.
- Let your Web page's user know they can stop sound from playing in your Web page by clicking the Stop button in their browsers.
- Enter the <embed> tag and a link to the sound file you want to use.
- An example looks like this: <C:\xampp\htdocs\play\collection”>, “D:\BOLLYWOOD” is a link to the sound file.

- The simplest way to be sure you have the link right is to place the sound file in the same folder as the Web page; that way the link is simply the filename.
- Click File→Save and reopen the file.
- The sound should play. Test the link right away to be sure it will work.
- If the sound doesn't play, experiment to make sure you have the path right and that sound plays on your machine.
- To make sure you have the link right, put the file in the same folder as your Web page and simplify the link. To make sure that sound playback works on your machine, navigate to the file in Windows Explorer and click it. It should play. If not, identify and fix the files affecting sound playback on your machine.

Result and Discussion

The result presented in this thesis project has utilized HTML5 technology to support multiple hardware platforms. Even though it is less neither stable nor compatible, as more and more major web browsers start to support or improve the current support of the Web Audio API audio engine as well as animation, the future of the development capability is brighter.

Conclusion

The Web-Based Music Player is used to automate and give a better music player experience for the end user. The application solves the basic needs of music listeners without troubling

them as existing applications do: it uses technology to increase the interaction of the system with the user in many ways. It eases the work of the end-user by capturing the image using a camera, determining their emotion, and suggesting a customized play-list through a more advanced and interactive system. The user will also be notified of songs that are not being played, to help them free up storage space .

References

- https://www.dummies.com/web-design-development/site-development/how-to-add-sound-to-your-web-site-using-html
- https://nevonprojects.com/media-player-project
- All music, 2017. [Online; accessed 11-July- 2017].
- Css - style sheet, 2017. [Online; accessed 11-July-2017]
- Java script, 2017. [Online; accessed 11-July-2017]
- Nodejs, 2017. [Online; accessed 11-July-2017].
- https://www.dummies.com/web-design-development/site-development/how-to-add-sound-to-your-web-site-using-html
- https://www.egrappler.com/wonderful-javascript-audio-libraries-for-developers
- https://nevonprojects.com/media-player-project
- https://pub.tik.ee.ethz.ch/students/2009-FS/SA-2009-10

Bossard, L. (2008). Pancho The Mobile Music Explorer. Zürich: DCG.TIK.EE.ETZH.CH